

MySQL and Windows

Abstract

This is the MySQL Windows extract from the MySQL 5.6 Reference Manual.

For legal information, see the [Legal Notices](#).

For help with using MySQL, please visit either the [MySQL Forums](#) or [MySQL Mailing Lists](#), where you can discuss your issues with other MySQL users.

For additional documentation on MySQL products, including translations of the documentation into other languages, and downloadable versions in variety of formats, including HTML and PDF formats, see the [MySQL Documentation Library](#).

Licensing information—MySQL 5.6. This product may include third-party software, used under license. If you are using a *Commercial* release of MySQL 5.6, see [this document](#) for licensing information, including licensing information relating to third-party software that may be included in this Commercial release. If you are using a *Community* release of MySQL 5.6, see [this document](#) for licensing information, including licensing information relating to third-party software that may be included in this Community release.

Licensing information—MySQL Cluster. This product may include third-party software, used under license. If you are using a *Commercial* release of MySQL Cluster NDB 7.3 or NDB 7.4, see [this document](#) for licensing information, including licensing information relating to third-party software that may be included in this Commercial release. If you are using a *Community* release of MySQL Cluster NDB 7.3 or NDB 7.4, see [this document](#) for licensing information, including licensing information relating to third-party software that may be included in this Community release.

Document generated on: 2016-08-04 (revision: 48445)

Table of Contents

Preface and Legal Notices	v
1 Installing MySQL on Microsoft Windows	1
1.1 MySQL Installation Layout on Microsoft Windows	3
1.2 Choosing An Installation Package	4
1.3 Installing MySQL on Microsoft Windows Using MySQL Installer	5
1.3.1 MySQL Installer GUI	7
1.3.2 MySQL Installer Console	30
1.4 MySQL Notifier	33
1.4.1 MySQL Notifier Usage	34
1.4.2 Setting Up Remote Monitoring in MySQL Notifier	41
1.5 Installing MySQL on Microsoft Windows Using a noinstall Zip Archive	46
1.5.1 Extracting the Install Archive	46
1.5.2 Creating an Option File	46
1.5.3 Selecting a MySQL Server Type	47
1.5.4 Starting the Server for the First Time	48
1.5.5 Starting MySQL from the Windows Command Line	49
1.5.6 Customizing the PATH for MySQL Tools	50
1.5.7 Starting MySQL as a Windows Service	51
1.5.8 Testing The MySQL Installation	53
1.6 Troubleshooting a Microsoft Windows MySQL Server Installation	54
1.7 Windows Postinstallation Procedures	56
1.8 Upgrading MySQL on Windows	58
2 Connection to MySQL Server Failing on Windows	61
3 Resetting the Root Password: Windows Systems	63
4 MySQL for Excel	65
5 Installation	67
6 Configuration	69
6.1 Global Options and Preferences	69
6.2 Managing MySQL Connections	73
7 What Is New In MySQL for Excel	77
7.1 What Is New In MySQL for Excel 1.3	77
7.2 What Is New In MySQL for Excel 1.2	77
8 Edit MySQL Data in Excel	79
9 Import MySQL Data into Excel	81
9.1 Choosing Columns To Export	81
9.2 Importing a Table	81
9.3 Import: Advanced Options	82
9.4 Importing a View or Procedure	84
9.5 Adding Summary Fields	85
9.6 Creating PivotTables	87
10 Append Excel Data into MySQL	97
11 Export Excel Data into MySQL	101
12 MySQL for Excel Frequently Asked Questions	105

Preface and Legal Notices

This is the MySQL Windows extract from the MySQL 5.6 Reference Manual.

Legal Notices

Copyright © 1997, 2016, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

This documentation is NOT distributed under a GPL license. Use of this documentation is subject to the following terms:

You may create a printed copy of this documentation solely for your own personal use. Conversion to other formats is allowed as long as the actual content is not altered or edited in any way. You shall not publish or distribute this documentation in any form or on any media, except if you distribute the documentation in a manner similar to how Oracle disseminates it (that is, electronically for download on a Web site with the software) or on a CD-ROM or similar medium, provided however that the documentation is disseminated together with the software on the same medium. Any other use, such as any dissemination of printed copies or use of this documentation, in whole or in part, in another publication, requires the prior written consent from an authorized representative of Oracle. Oracle and/or its affiliates reserve any and all rights to this documentation not expressly granted above.

Chapter 1 Installing MySQL on Microsoft Windows

Table of Contents

1.1 MySQL Installation Layout on Microsoft Windows	3
1.2 Choosing An Installation Package	4
1.3 Installing MySQL on Microsoft Windows Using MySQL Installer	5
1.3.1 MySQL Installer GUI	7
1.3.2 MySQL Installer Console	30
1.4 MySQL Notifier	33
1.4.1 MySQL Notifier Usage	34
1.4.2 Setting Up Remote Monitoring in MySQL Notifier	41
1.5 Installing MySQL on Microsoft Windows Using a noinstall Zip Archive	46
1.5.1 Extracting the Install Archive	46
1.5.2 Creating an Option File	46
1.5.3 Selecting a MySQL Server Type	47
1.5.4 Starting the Server for the First Time	48
1.5.5 Starting MySQL from the Windows Command Line	49
1.5.6 Customizing the PATH for MySQL Tools	50
1.5.7 Starting MySQL as a Windows Service	51
1.5.8 Testing The MySQL Installation	53
1.6 Troubleshooting a Microsoft Windows MySQL Server Installation	54
1.7 Windows Postinstallation Procedures	56
1.8 Upgrading MySQL on Windows	58

There are several different methods to install MySQL on Microsoft Windows.

Simple Installation Method

The simplest and recommended method is to download MySQL Installer (for Windows) and let it install and configure all of the MySQL products on your system. Here is how:

- Download MySQL Installer from <http://dev.mysql.com/downloads/installer/> and execute it.

Note

Unlike the standard MySQL Installer, the smaller "web-community" version does not bundle any MySQL applications but it will download the MySQL products you choose to install.

- Choose the appropriate **Setup Type** for your system. Typically you will choose **Developer Default** to install MySQL server and other MySQL tools related to MySQL development, helpful tools like MySQL Workbench. Or, choose the **Custom** setup type to manually select your desired MySQL products.

Note

Multiple versions of MySQL server can exist on a single system. You can choose one or multiple versions.

- Complete the installation process by following the MySQL Installation wizard's instructions. This will install several MySQL products and start the MySQL server.
- MySQL is now installed. You probably configured MySQL as a service that will automatically start MySQL server every time you restart your system.

Note

You probably also installed other helpful MySQL products like MySQL Workbench and MySQL Notifier on your system. Consider loading [MySQL Workbench](#) to check your new MySQL server connection, and [Section 1.4, “MySQL Notifier”](#) to view the connection's status. By default, these two programs automatically start after installing MySQL.

This process also installs the MySQL Installer application on your system, and later you can use MySQL Installer to upgrade or reconfigure your MySQL products.

Additional Installation Information

MySQL is available for Microsoft Windows, for both 32-bit and 64-bit versions. For supported Windows platform information, see <http://www.mysql.com/support/supportedplatforms/database.html>.

It is possible to run MySQL as a standard application or as a Windows service. By using a service, you can monitor and control the operation of the server through the standard Windows service management tools. For more information, see [Section 1.5.7, “Starting MySQL as a Windows Service”](#).

Generally, you should install MySQL on Windows using an account that has administrator rights. Otherwise, you may encounter problems with certain operations such as editing the [PATH](#) environment variable or accessing the [Service Control Manager](#). Once installed, MySQL does not need to be executed using a user with Administrator privileges.

For a list of limitations on the use of MySQL on the Windows platform, see [Windows Platform Limitations](#).

In addition to the MySQL Server package, you may need or want additional components to use MySQL with your application or development environment. These include, but are not limited to:

- To connect to the MySQL server using ODBC, you must have a Connector/ODBC driver. For more information, including installation and configuration instructions, see [MySQL Connector/ODBC Developer Guide](#).

Note

MySQL Installer will install and configure Connector/ODBC for you.

- To use MySQL server with .NET applications, you must have the Connector/Net driver. For more information, including installation and configuration instructions, see [MySQL Connector/Net Developer Guide](#).

Note

MySQL Installer will install and configure Connector/NET for you.

MySQL distributions for Windows can be downloaded from <http://dev.mysql.com/downloads/>. See [How to Get MySQL](#).

MySQL for Windows is available in several distribution formats, detailed here. Generally speaking, you should use MySQL Installer. It contains more features and MySQL products than the older MSI, is simpler to use than the Zip file, and you need no additional tools to get MySQL up and running. MySQL Installer automatically installs MySQL Server and additional MySQL products, creates an options file, starts the server, and enables you to create default user accounts. For more information on choosing a package, see [Section 1.2, “Choosing An Installation Package”](#).

- A MySQL Installer distribution includes MySQL Server and additional MySQL products including MySQL Workbench, MySQL Notifier, and MySQL for Excel. MySQL Installer can also be used to upgrade these products in the future.

For instructions on installing MySQL using MySQL Installer, see [Section 1.3, “Installing MySQL on Microsoft Windows Using MySQL Installer”](#).

- The standard binary distribution (packaged as a Zip file) contains all of the necessary files that you unpack into your chosen location. This package contains all of the files in the full Windows MSI Installer package, but does not include an installation program.

For instructions on installing MySQL using the Zip file, see [Section 1.5, “Installing MySQL on Microsoft Windows Using a noinstall Zip Archive”](#).

- The source distribution format contains all the code and support files for building the executables using the Visual Studio compiler system.

For instructions on building MySQL from source on Windows, see [Installing MySQL from Source](#).

MySQL on Windows considerations:

- **Large Table Support**

If you need tables with a size larger than 4GB, install MySQL on an NTFS or newer file system. Do not forget to use `MAX_ROWS` and `AVG_ROW_LENGTH` when you create tables. See [CREATE TABLE Syntax](#).

- **MySQL and Virus Checking Software**

Virus-scanning software such as Norton/Symantec Anti-Virus on directories containing MySQL data and temporary tables can cause issues, both in terms of the performance of MySQL and the virus-scanning software misidentifying the contents of the files as containing spam. This is due to the fingerprinting mechanism used by the virus-scanning software, and the way in which MySQL rapidly updates different files, which may be identified as a potential security risk.

After installing MySQL Server, it is recommended that you disable virus scanning on the main directory (`datadir`) used to store your MySQL table data. There is usually a system built into the virus-scanning software to enable specific directories to be ignored.

In addition, by default, MySQL creates temporary files in the standard Windows temporary directory. To prevent the temporary files also being scanned, configure a separate temporary directory for MySQL temporary files and add this directory to the virus scanning exclusion list. To do this, add a configuration option for the `tmpdir` parameter to your `my.ini` configuration file. For more information, see [Section 1.5.2, “Creating an Option File”](#).

1.1 MySQL Installation Layout on Microsoft Windows

For MySQL 5.6 on Windows, the default installation directory is `C:\Program Files\MySQL\MySQL Server 5.6`. Some Windows users prefer to install in `C:\mysql`, the directory that formerly was used as the default. However, the layout of the subdirectories remains the same.

All of the files are located within this parent directory, using the structure shown in the following table.

Table 1.1 Default MySQL Installation Layout for Microsoft Windows

Directory	Contents of Directory	Notes
<code>bin, scripts</code>	<code>mysqld</code> server, client and utility programs	

Directory	Contents of Directory	Notes
%ALLUSERSPROFILE%\MySQL\MySQL Server 5.6\	Log files, databases (Windows XP, Windows Server 2003)	The Windows system variable %ALLUSERSPROFILE% defaults to C:\Documents and Settings\All Users\Application Data
%PROGRAMDATA%\MySQL\MySQL Server 5.6\	Log files, databases (Vista, Windows 7, Windows Server 2008, and newer)	The Windows system variable %PROGRAMDATA% defaults to C:\ProgramData
examples	Example programs and scripts	
include	Include (header) files	
lib	Libraries	
share	Miscellaneous support files, including error messages, character set files, sample configuration files, SQL for database installation	

If you install MySQL using the MySQL Installer, this package creates and sets up the data directory that the installed server will use, and also creates a pristine “template” data directory named [data](#) under the installation directory. After an installation has been performed using this package, the template data directory can be copied to set up additional MySQL instances. See [Running Multiple MySQL Instances on One Machine](#).

1.2 Choosing An Installation Package

For MySQL 5.6, there are multiple installation package formats to choose from when installing MySQL on Windows.

Note

Program Database (PDB) files (with file name extension [pdb](#)) provide information for debugging your MySQL installation in the event of a problem. These files are included in ZIP Archive distributions (but not MSI distributions) of MySQL.

- **MySQL Installer:** This package has a file name similar to [mysql-installer-community-5.6.33.0.msi](#) or [mysql-installer-commercial-5.6.33.0.msi](#), and utilizes MSIs to automatically install MySQL server and other products. It will download and apply updates to itself, and for each of the installed products. It also configures the additional non-server products.

The installed products are configurable, and this includes: documentation with samples and examples, connectors (such as C, C++, J, NET, and ODBC), MySQL Workbench, MySQL Notifier, MySQL for Excel, and the MySQL Server with its components.

MySQL Installer will run on all Windows platforms that are supported by MySQL (see <http://www.mysql.com/support/supportedplatforms/database.html>).

Note

Because MySQL Installer is not a native component of Microsoft Windows and depends on .NET, it will not work on minimal installation options like the “Server Core” version of Windows Server 2008.

For instructions on installing MySQL using MySQL Installer, see [Section 1.3, “Installing MySQL on Microsoft Windows Using MySQL Installer”](#).

- **The Noinstall Archive:** This package has a file name similar to `mysql-5.6.33-win32.zip` or `mysql-5.6.33-win64.zip`, and contains all the files found in the Complete install package, with the exception of the GUI. It also contains PDB files. This package does not include an automated installer, and must be manually installed and configured.

MySQL Installer is recommended for most users.

Your choice of install package affects the installation process you must follow. If you choose to use MySQL Installer, see [Section 1.3, “Installing MySQL on Microsoft Windows Using MySQL Installer”](#). If you choose to install a Noinstall archive, see [Section 1.5, “Installing MySQL on Microsoft Windows Using a noinstall Zip Archive”](#).

1.3 Installing MySQL on Microsoft Windows Using MySQL Installer

MySQL Installer is an application that manages MySQL products on Microsoft Windows. It installs, updates, removes, and configures MySQL products, and remains on the system as its own application. MySQL Installer is only available for Microsoft Windows, and includes both GUI and command-line interfaces.

The supported MySQL products include:

- [MySQL Server](#) (one or multiple versions on the same system)
- [MySQL Workbench](#)
- [MySQL Connectors](#) (.Net / Python / ODBC / Java / C / C++)
- [MySQL Notifier](#)
- [MySQL for Excel](#)
- [MySQL for Visual Studio](#)
- [MySQL Utilities and MySQL Fabric](#)
- MySQL Samples and Examples
- MySQL Documentation
- MySQL Installer is also installed and remains on the system as its own application, that is used to install additional MySQL products, and also to update and configure existing MySQL products
- The Enterprise edition installs the Enterprise versions of the above products, and also includes MySQL Enterprise Backup and MySQL Enterprise Firewall

Installer package types

- **Full:** Bundles all of the MySQL products (including the MySQL server). The file size is over 300MB, and its name has the form `mysql-installer-community-VERSION.N.msi` where `VERSION` is the MySQL Server version number such as 5.7 and `N` is the package number, which begins at 0.
- **Web:** Only contains the Installer and configuration files, and it downloads the MySQL products you choose to install. The size of this file is about 2MB; the name of the file has the form `mysql-`

`installer-community-web-VERSION.N.msi` where `VERSION` is the MySQL Server version number such as 5.7 and `N` is the package number, which begins at 0.

- **Updates:** MySQL Installer can upgrade itself, so an additional download is not required to update MySQL Installer.

Installer editions

- **Community edition:** Downloadable at <http://dev.mysql.com/downloads/installer/>. It installs the community edition of all MySQL products.
- **Commercial edition:** Downloadable at either [My Oracle Support \(MOS\)](#) or <https://edelivery.oracle.com/>. It installs the commercial version of all MySQL products, including Workbench SE/EE, MySQL Enterprise Backup, and MySQL Enterprise Firewall. It also integrates with your MOS account.

Note

Entering your MOS credentials is optional when installing bundled MySQL products, but your credentials are required when choosing non-bundled MySQL products that MySQL Installer must download.

For notes detailing the changes in each release of MySQL Installer, see [MySQL Installer Release Notes](#).

MySQL Installer is compatible with pre-existing installations, and adds them to its list of installed components. While the standard MySQL Installer is bundled with a specific version of MySQL server, a single MySQL Installer instance can install and manage multiple MySQL server versions. For example, a single MySQL Installer instance can install (and update) versions 5.5, 5.6, and 5.7 on the same host.

Note

A single host can *not* have both community and commercial editions of MySQL server installed. For example, if you want both MySQL Server 5.6 and 5.7 installed on a single host, both must be the same edition.

MySQL Installer handles the initial configuration and set up of the applications. For example:

1. It creates the configuration file (`my.ini`) that is used to configure the MySQL Server. The values written to this file are influenced by choices you make during the installation process.

Note

Some definitions are host dependent. For example, `query_cache` is enabled if the host has fewer than three cores.

2. It can optionally import example databases.
3. By default, a Windows service for the MySQL server is added.
4. It can optionally create MySQL Server user accounts with configurable permissions based on general roles, such as DB Administrator, DB Designer, and Backup Admin. It optionally creates a Windows user named `MySQLSys` with limited privileges, which would then run the MySQL Server.

User accounts may also be added and configured in MySQL Workbench.

5. Checking **Show Advanced Options** allows additional **Logging Options** to be set. This includes defining custom file paths for the error log, general log, slow query log (including the configuration of seconds it requires to execute a query), and the binary log.

MySQL Installer can optionally check for updated components and download them for you.

1.3.1 MySQL Installer GUI

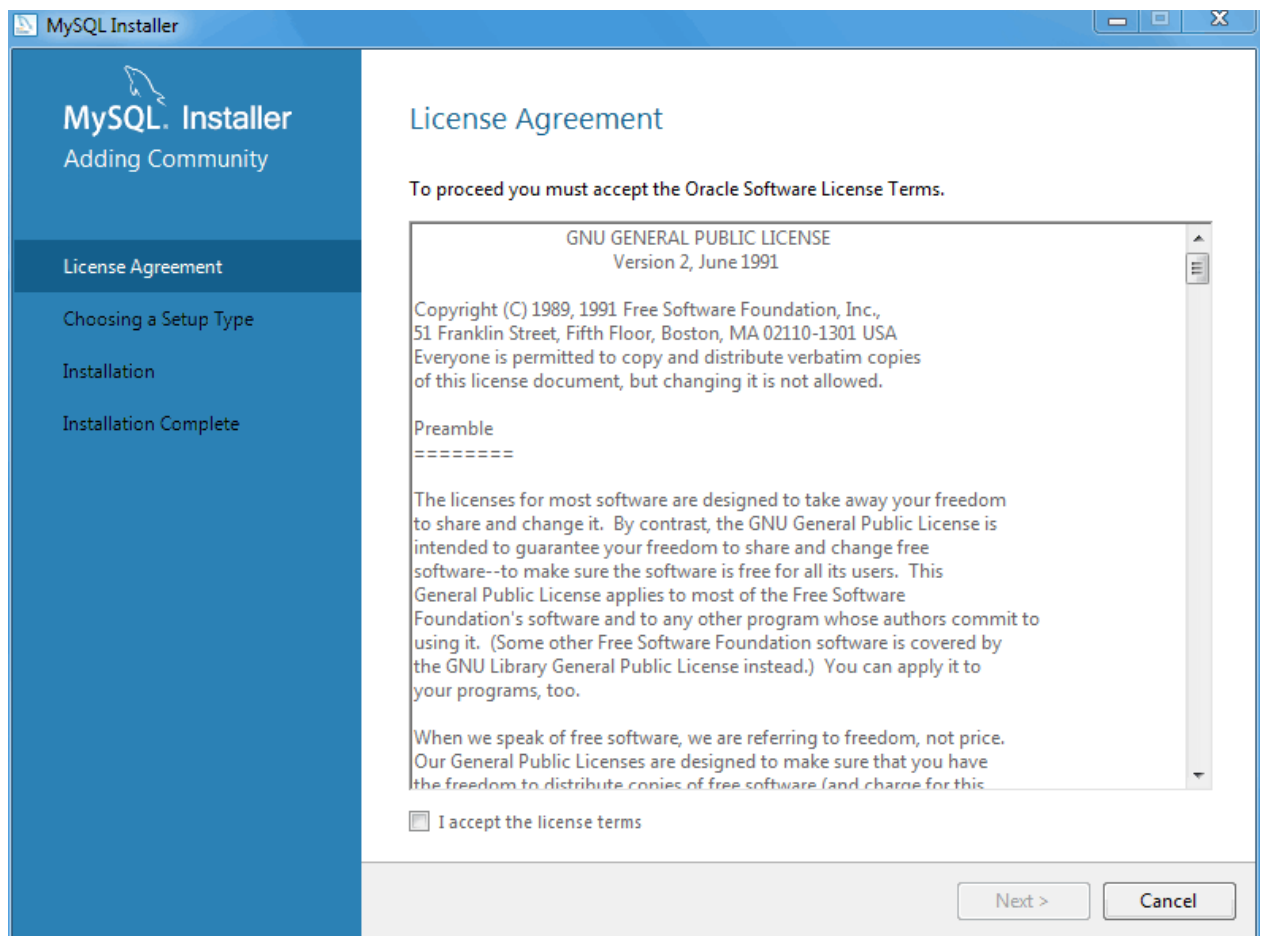
Installing MySQL Installer adds a link to the Start menu under the **MySQL** group. Click **Start, All Programs, MySQL, MySQL Installer** to reload the MySQL Installer GUI.

Note

Full permissions are granted to the user executing MySQL Installer to all generated files, such as `my.ini`. This does not apply to files and directories for specific products, such as the MySQL server data directory in `%ProgramData%` that is owned by `SYSTEM`.

MySQL Installer requires you to accept the license agreement before it will install MySQL products.

Figure 1.1 MySQL Installer - License Agreement



Installing New Packages

Choose the appropriate **Setup Type** for your system. This type determines which MySQL products are initially installed on your system, or select **Custom** to manually choose the products.

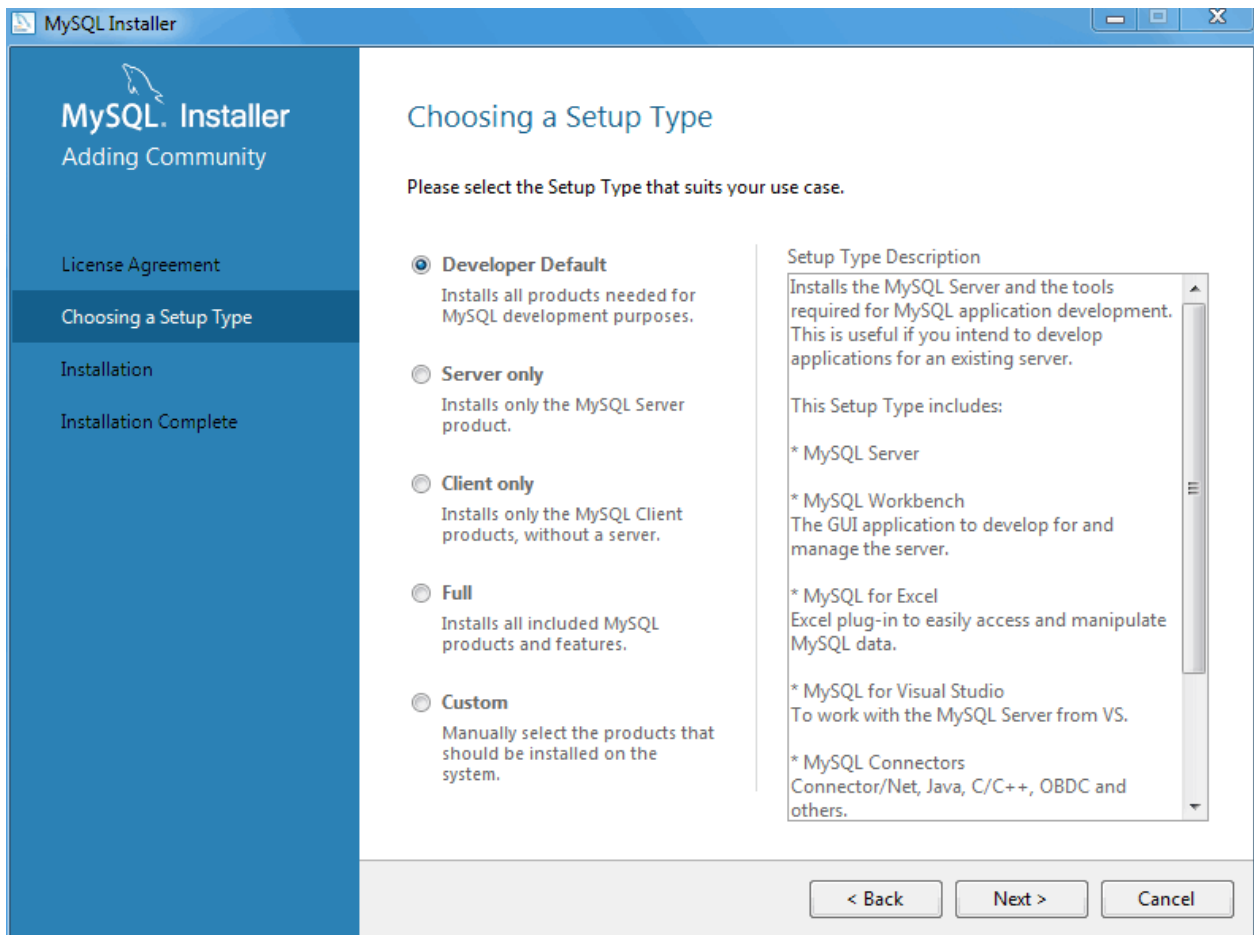
- **Developer:** Install all products needed to develop applications with MySQL. This is the default option.

- **Server only:** Only install the MySQL server.
- **Client only:** Only install the MySQL client products, such as MySQL Workbench. This does not include the MySQL server.
- **Full:** Install all available MySQL products.
- **Custom:** Manually select the MySQL products to install, and optionally configure custom MySQL data and installation paths.

Note

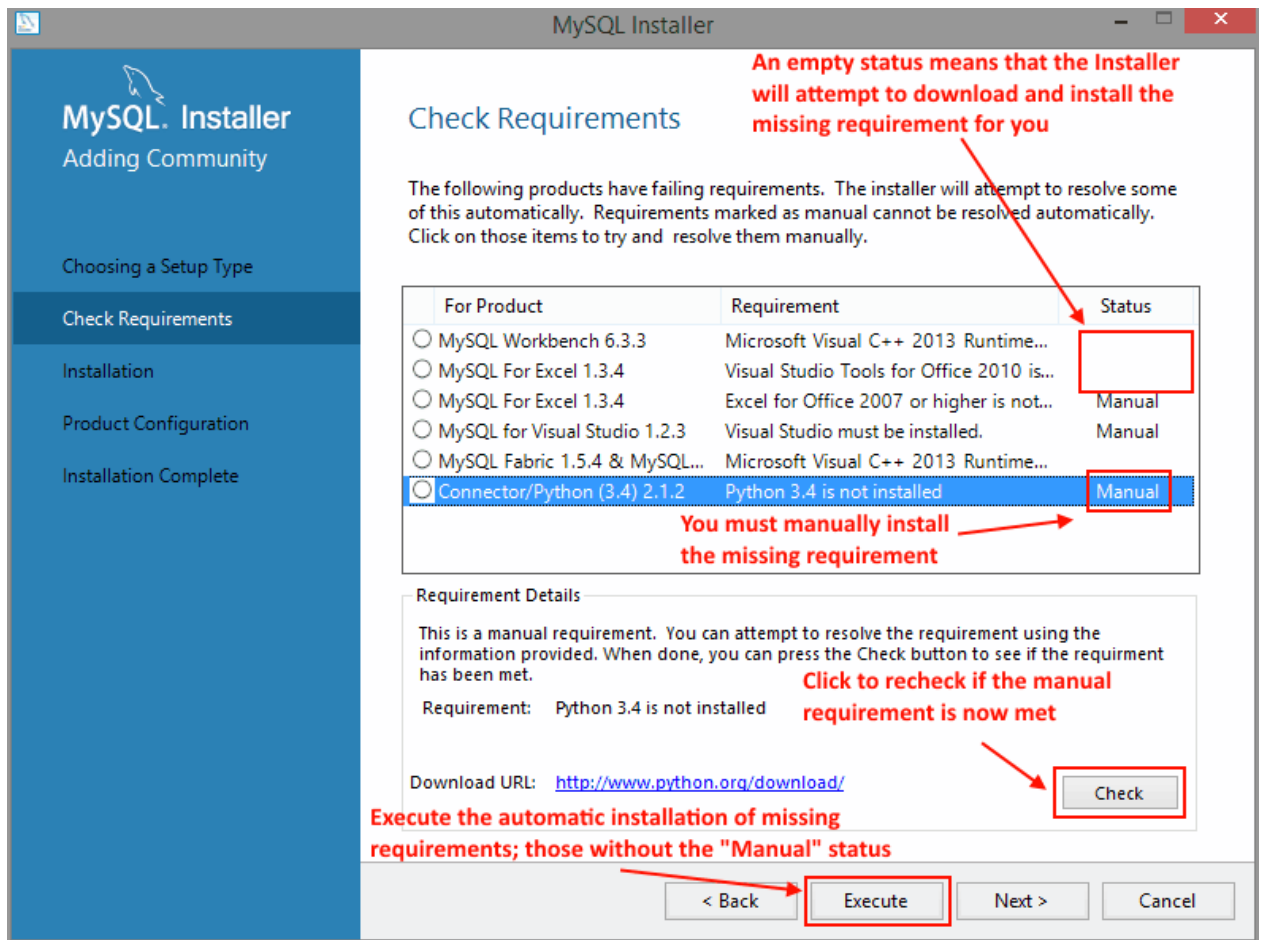
After the initial installation, you may use MySQL Installer to manually select MySQL products to install or remove. In other words, MySQL Installer becomes a MySQL product management system.

Figure 1.2 MySQL Installer - Choosing a Setup Type

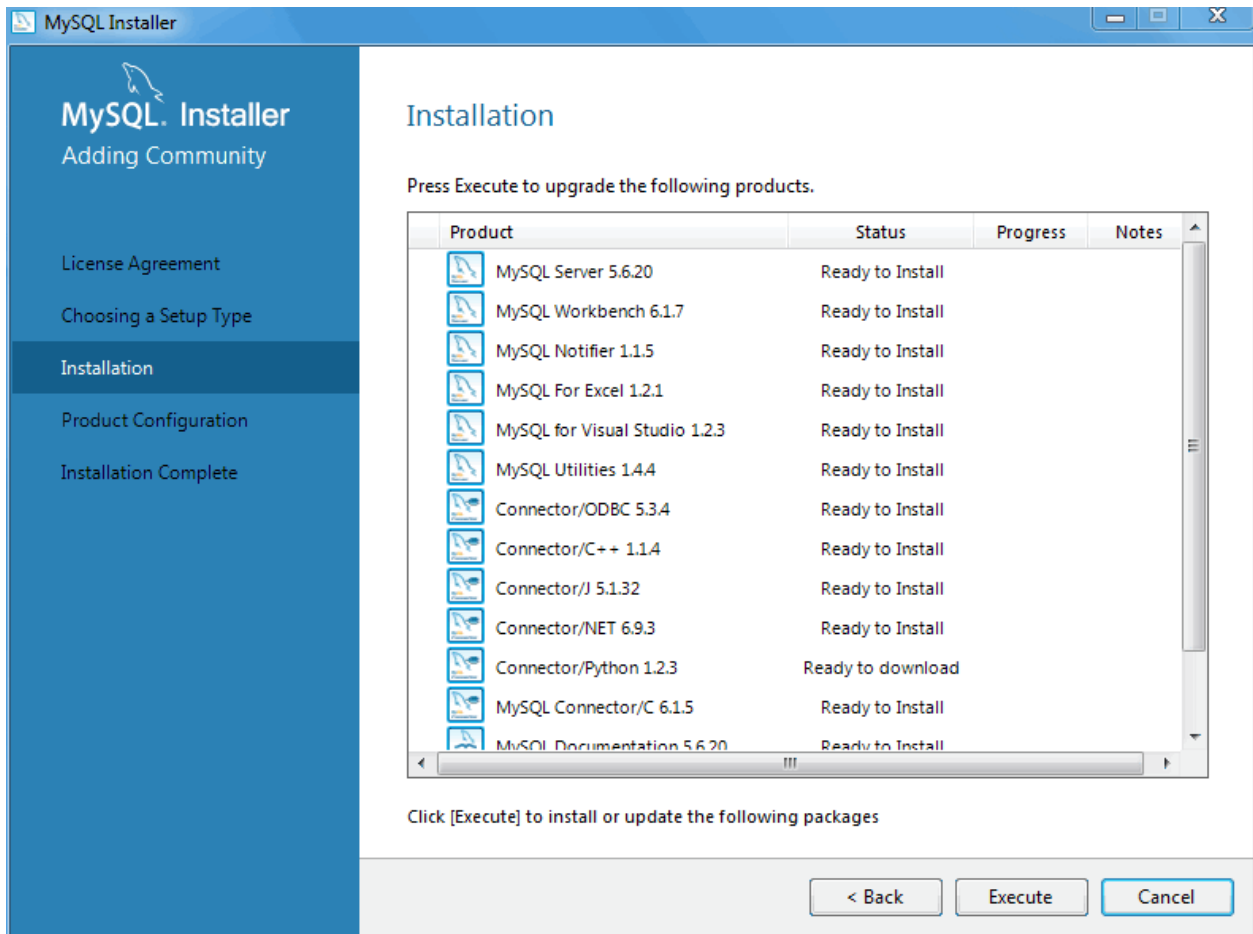


MySQL Installer checks your system for the external requirements (pre-requisites) required to install the selected MySQL products. MySQL Installer can download and install some prerequisites, but others require manual intervention. Download and install all prerequisites that have **Status** set to "Manual". Click **Check** to recheck if a manual prerequisite was installed. After manually installing those requirements, click **Execute** to download and install the other prerequisites. Once finished, click **Next** to continue.

Figure 1.3 MySQL Installer - Check Requirements



The next window lists the MySQL products that are scheduled for installation:

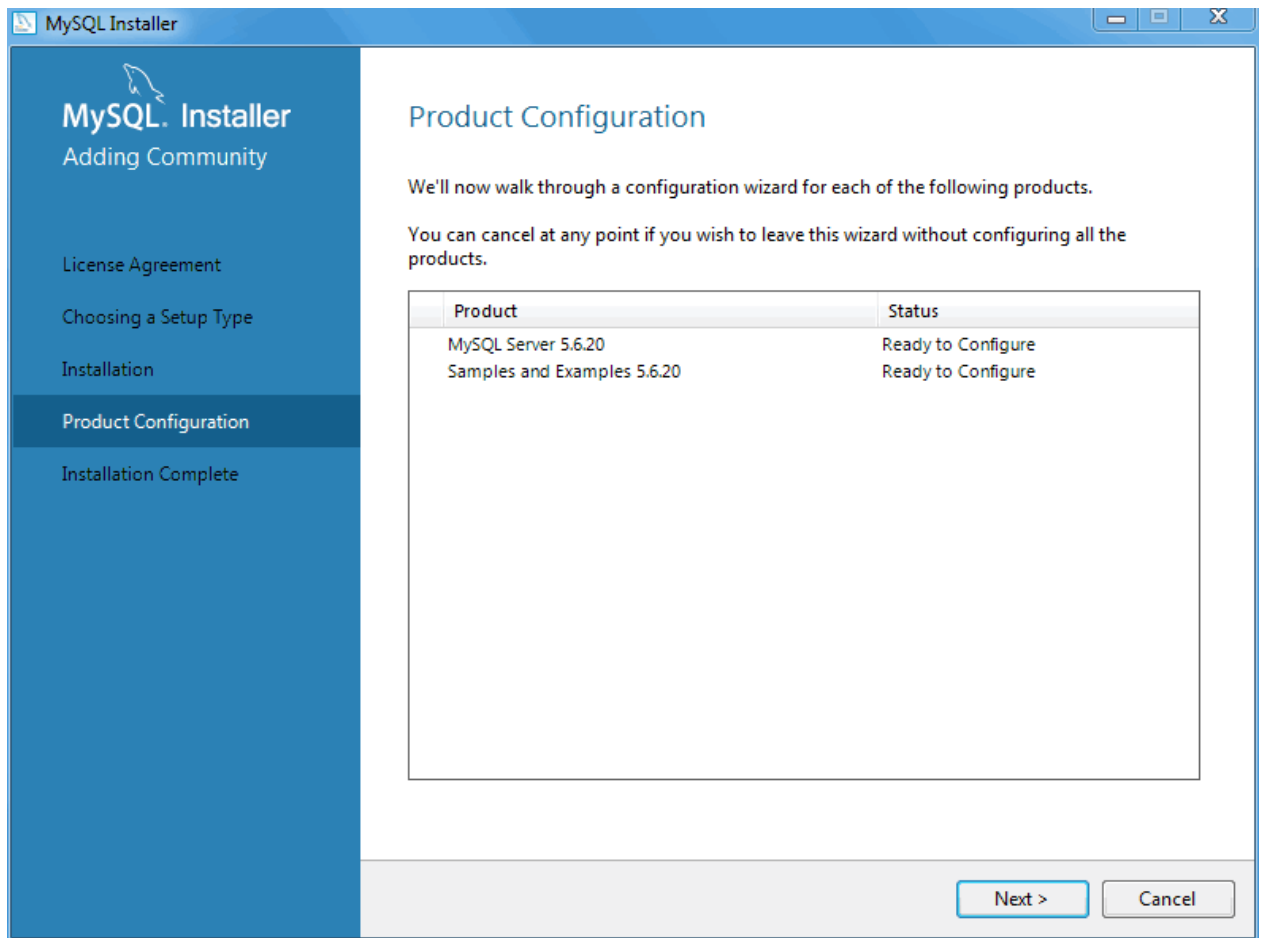
Figure 1.4 MySQL Installer - Installation Progress

As components are installed, their **Status** changes from a progress percentage to "Complete".

After all components are installed, the next step configures some of the recently installed MySQL products. The [Configuration Overview](#) window displays the progress and then loads a configuration window, if required. Our example configures MySQL Server 5.6.x.

Configuring MySQL Server

Configuring the MySQL server begins with defining several **Type and Networking** options.

Figure 1.5 MySQL Installer - Configuration Overview**Server Configuration Type**

Choose the MySQL server configuration type that describes your setup. This setting defines the amount of system resources (memory) that will be assigned to your MySQL server instance.

- **Developer:** A machine that will host many other applications, and typically this is your personal workstation. This option configures MySQL to use the least amount of memory.
- **Server:** Several other applications will be running on this machine, such as a web server. This option configures MySQL to use a medium amount of memory.
- **Dedicated:** A machine that is dedicated to running the MySQL server. Because no other major applications will run on this server, such as a web server, this option configures MySQL to use the majority of available memory.

Connectivity

Connectivity options control how the connection to MySQL is made. Options include:

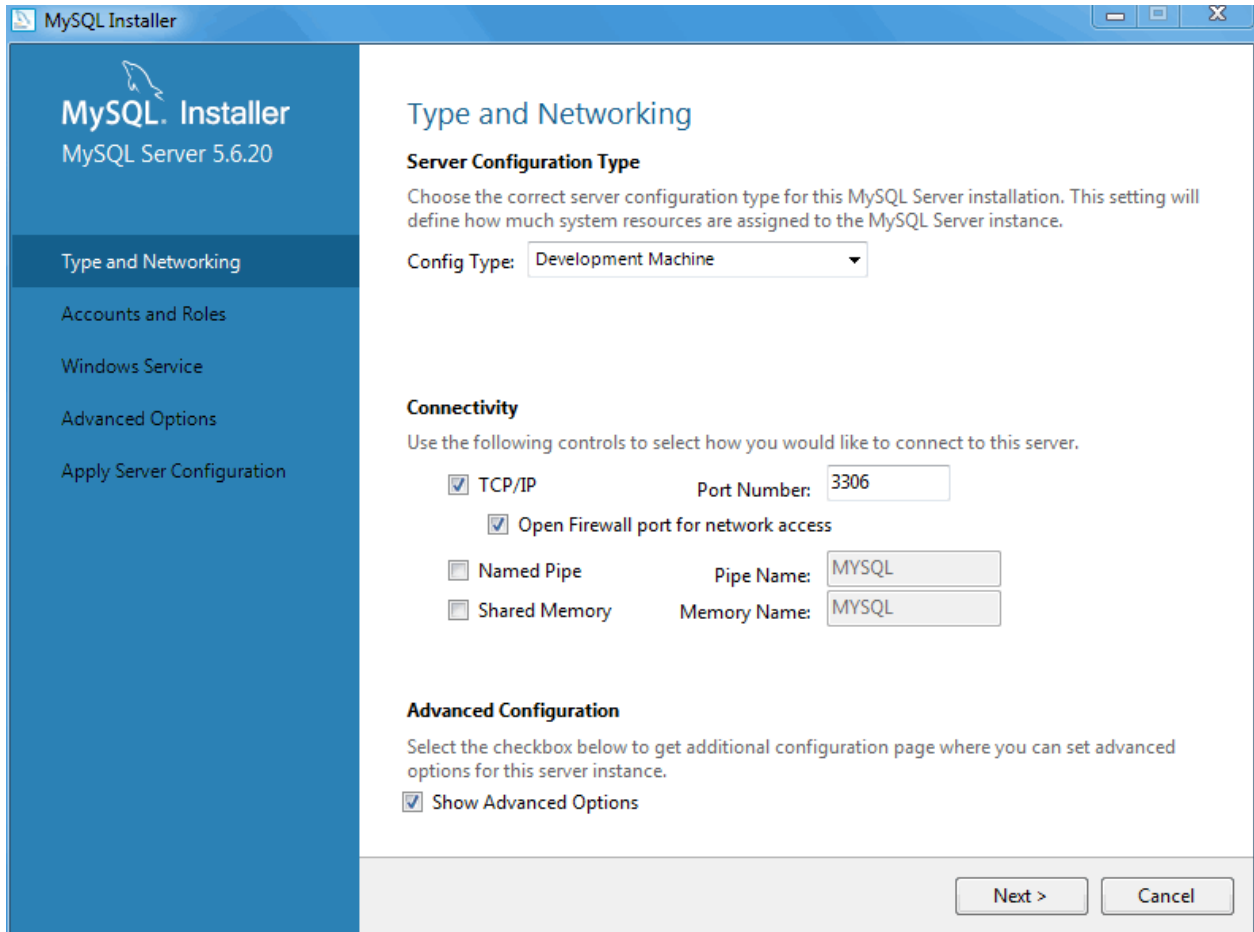
- **TCP/IP:** You may enable TCP/IP Networking here as otherwise only localhost connections are allowed. Also define the **Port Number** and whether to open the firewall port for network access.
- **Named Pipe:** Enable and define the pipe name, similar to using the `--enable-named-pipe` option.

- **Shared Memory:** Enable and then define the memory name, similar to using the `--shared-memory` option.

Advanced Configuration

Check **Show Advanced Options** to set additional **Logging Options**. This includes defining custom file paths for the error log, general log, slow query log (including the configuration of seconds it requires to execute a query), and the binary log.

Figure 1.6 MySQL Installer - MySQL Server Configuration: Type and Networking



Accounts and Roles

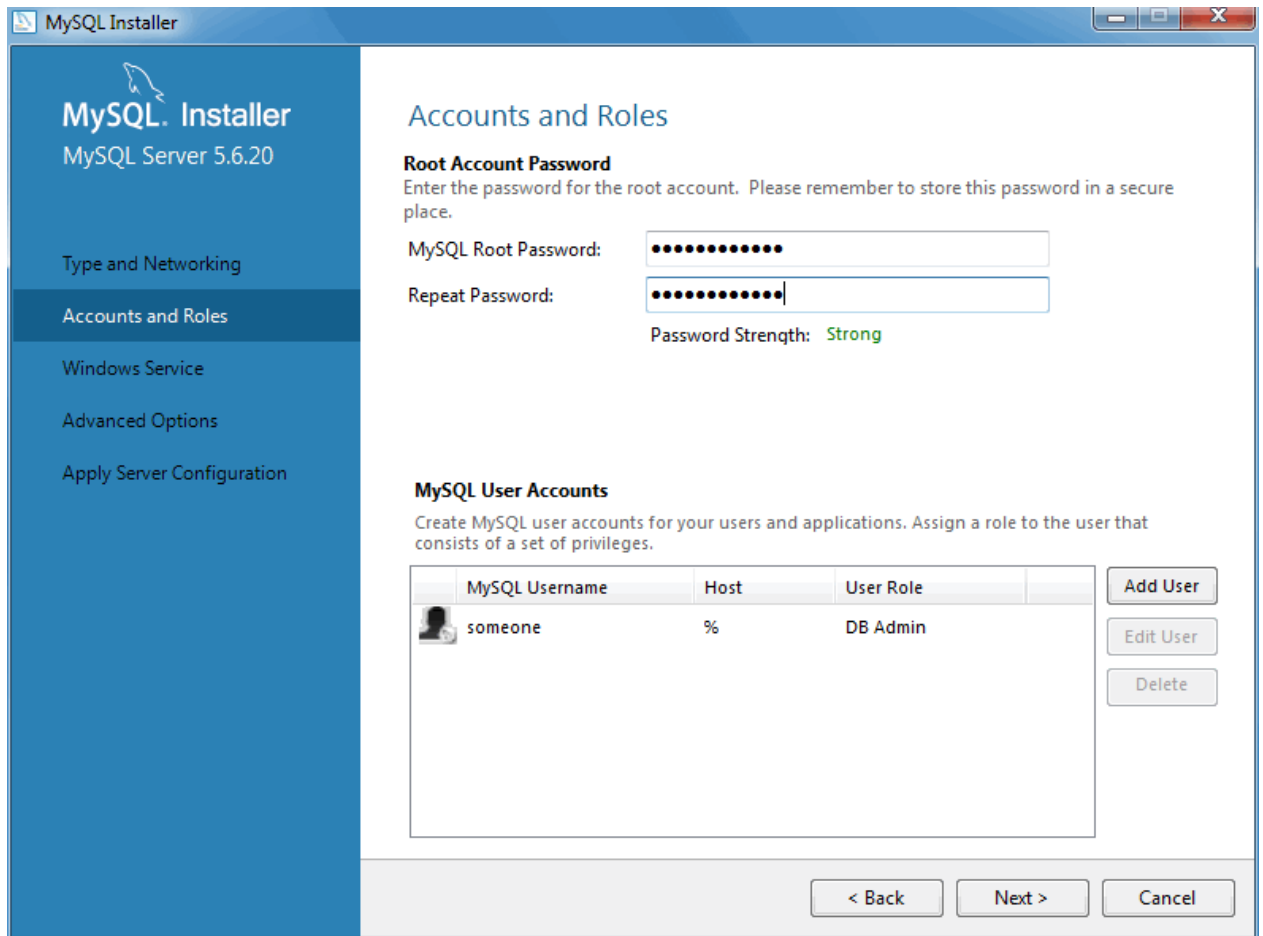
Next, define your MySQL account information. Assigning a root password is required.

Optionally, you can add additional MySQL user accounts with predefined user roles. Each predefined role, such as "DB Admin", are configured with their own set of privileges. For example, the "DB Admin" role has more privileges than the "DB Designer" role. Click the **Role** dropdown for a list of role descriptions.

Note

If the MySQL Server is already installed, then you must also enter the **Current Root Password**.

Figure 1.7 MySQL Installer - MySQL Server Configuration: User Accounts and Roles



The screenshot shows the 'Accounts and Roles' step of the MySQL Installer. The left sidebar contains the following options: 'Type and Networking', 'Accounts and Roles' (selected), 'Windows Service', 'Advanced Options', and 'Apply Server Configuration'. The main area is titled 'Accounts and Roles' and contains two sections:

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password: [password field]
Repeat Password: [password field]
Password Strength: **Strong**

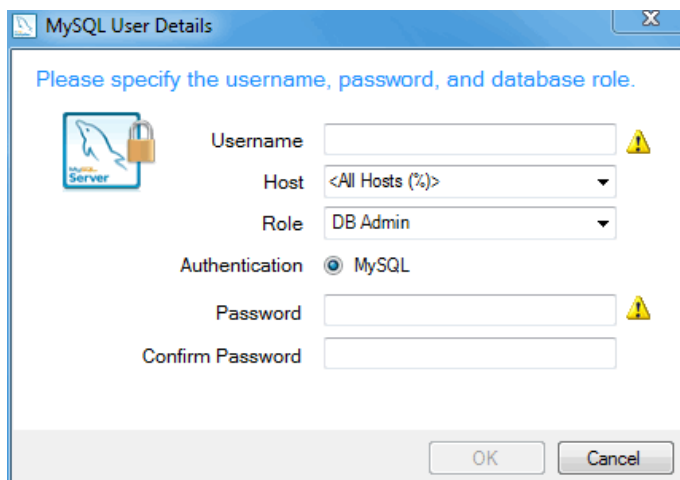
MySQL User Accounts
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL Username	Host	User Role
 someone	%	DB Admin

Buttons on the right: Add User, Edit User, Delete.

Buttons at the bottom: < Back, Next >, Cancel.

Figure 1.8 MySQL Installer - MySQL Server Configuration: User Accounts and Roles: Adding a User



The screenshot shows the 'MySQL User Details' dialog box. It contains the following fields and options:

Please specify the username, password, and database role.

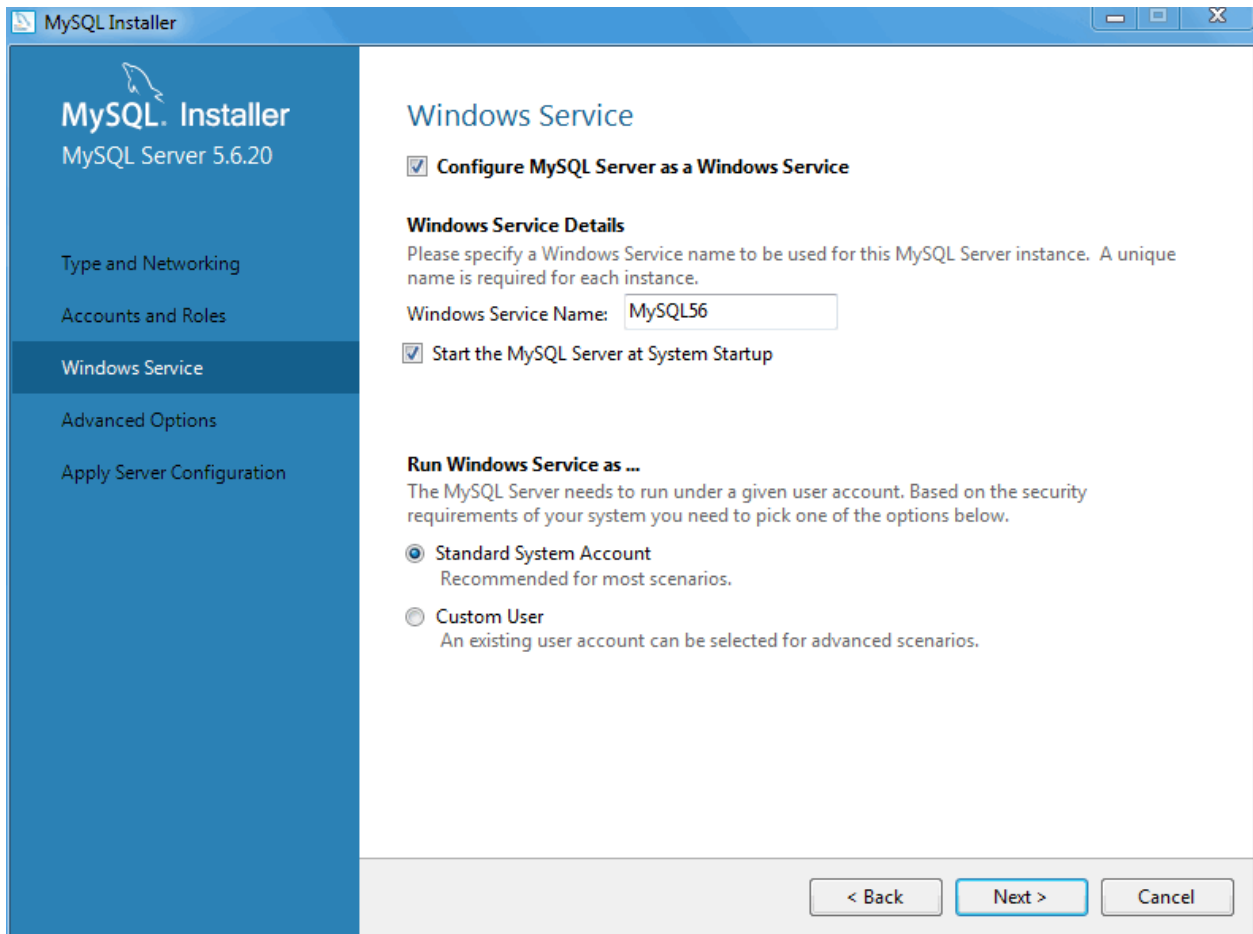
Username: [text field] ⚠️
Host: <All Hosts (%)> [dropdown]
Role: DB Admin [dropdown]
Authentication: ☒ MySQL
Password: [password field] ⚠️
Confirm Password: [password field]

Buttons at the bottom: OK, Cancel.

Windows Service

Next, configure the **Windows Service** details. This includes the service name, whether the MySQL server should be loaded at startup, and how the MySQL server Windows service is executed.

Figure 1.9 MySQL Installer - MySQL Server Configuration: Windows Service

**Note**

When configuring **Run Windows Services as ...** using a **Custom User**, the custom user must have privileges to log on to Microsoft Windows as a service. The **Next** button will be disabled until this user is configured with the required privileges.

On Microsoft Windows 7, this is configured by loading the **Start Menu, Control Panel, Administrative Tools, Local Security Policy, Local Policies, User Rights Assignment**, then **Log On As A Service**. Choose **Add User or Group** here to add the custom user, and then **OK, OK** to save.

Advanced Options

The next configuration step is available if the **Advanced Configuration** option was checked. This section includes options that are related to the MySQL log files:

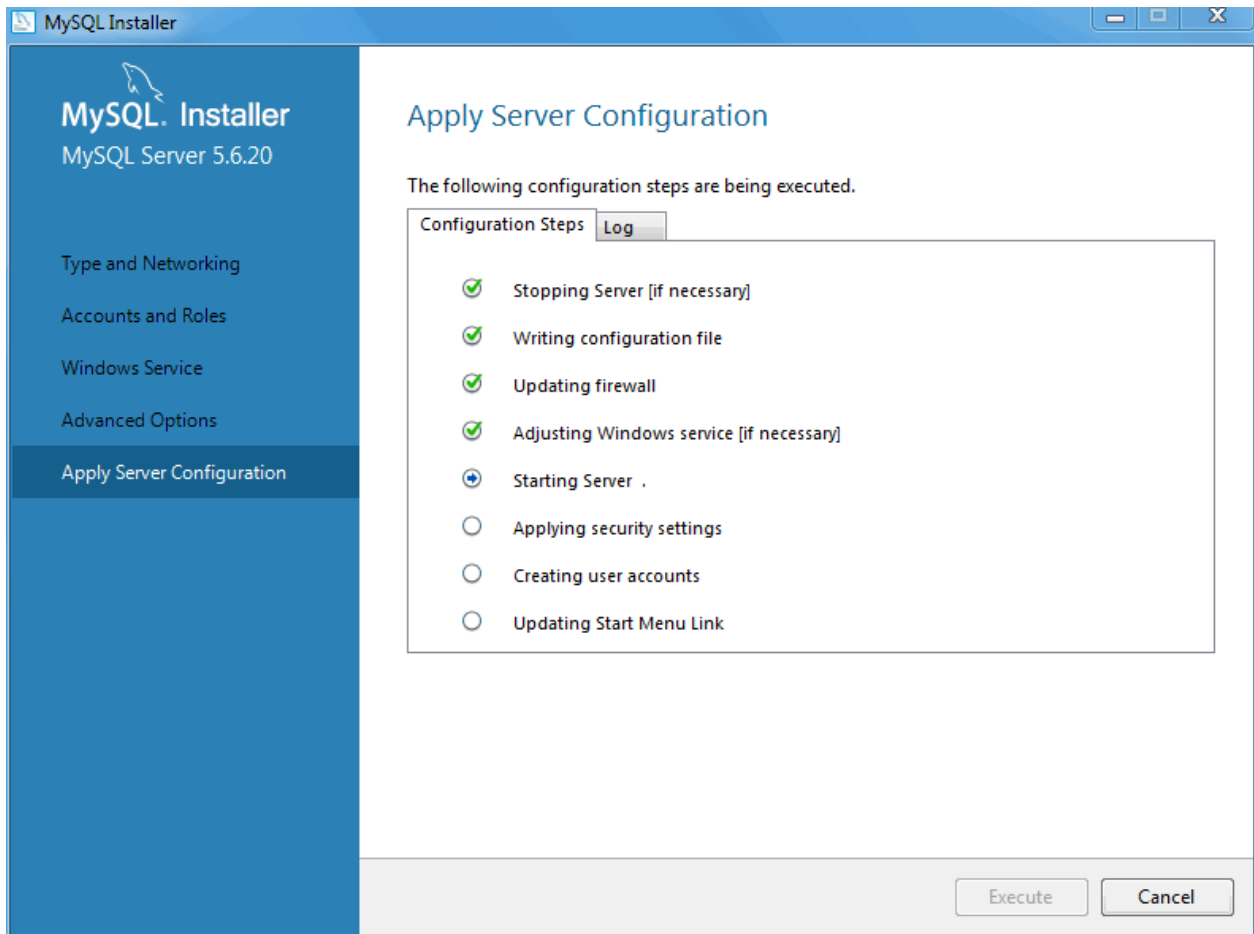
Figure 1.10 MySQL Installer - MySQL Server Configuration: Logging Options

The screenshot shows the MySQL Installer window for MySQL Server 5.6.20. The left sidebar contains the following navigation items: Type and Networking, Accounts and Roles, Windows Service, **Advanced Options** (selected), and Apply Server Configuration. The main content area is titled "Advanced Options" and "Logging Options". It includes a description: "Please select the logs you want to activate for this server in addition to the Error Log. On production machines it can be beneficial to separate the log files from the data." The configuration options are as follows:

- Error Log:** my-errors.err
- ☒ **General Log**
The general query log is a general record of what the MySQL Server is doing. It should only be used to track down issues.
File Path: my-general.log
- ☒ **Slow Query Log**
The slow query log consists of SQL statements that took more than the given value of seconds to execute. It is recommended to turn this log on.
File Path: my-slow.log **Seconds:** 10
- ☒ **Bin Log**
The binary log contains all database events and is used for replication and data recovery operations. It has a performance impact on the server. Only a filename base should be given as the server will append an appropriate extension.
File Path: my-bin
- Server Id:** 1

At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

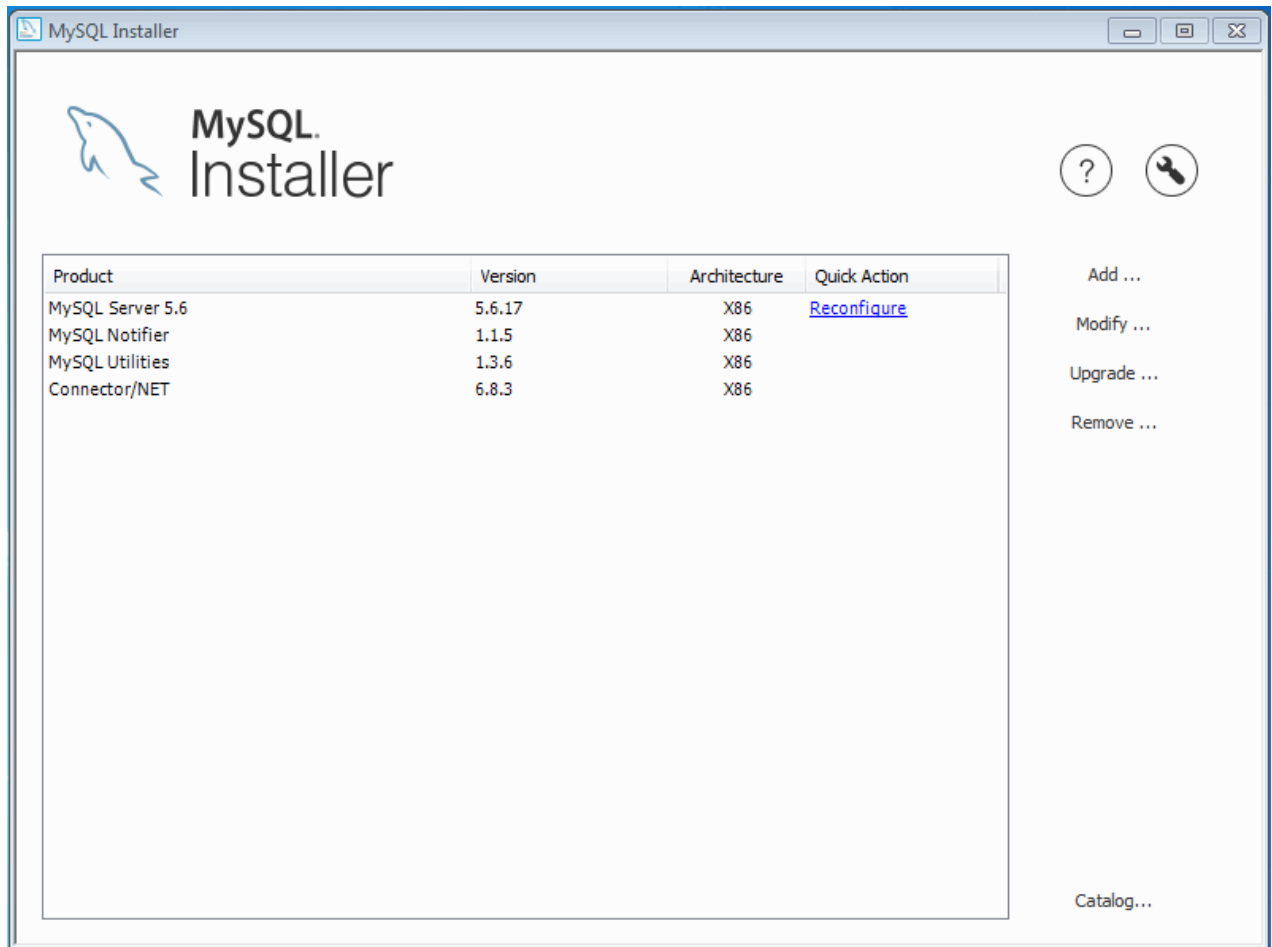
Click **Next** to continue on to the final page before all of the requested changes are applied. This **Apply Server Configuration** page details the configuration steps that will be performed.

Figure 1.11 MySQL Installer - MySQL Server Configuration: Apply Server Configuration

Click **Execute** to execute the configuration steps. The icon for each step toggles from white to green on success, or the process stops on failure. Click the **Log** tab to view the log.

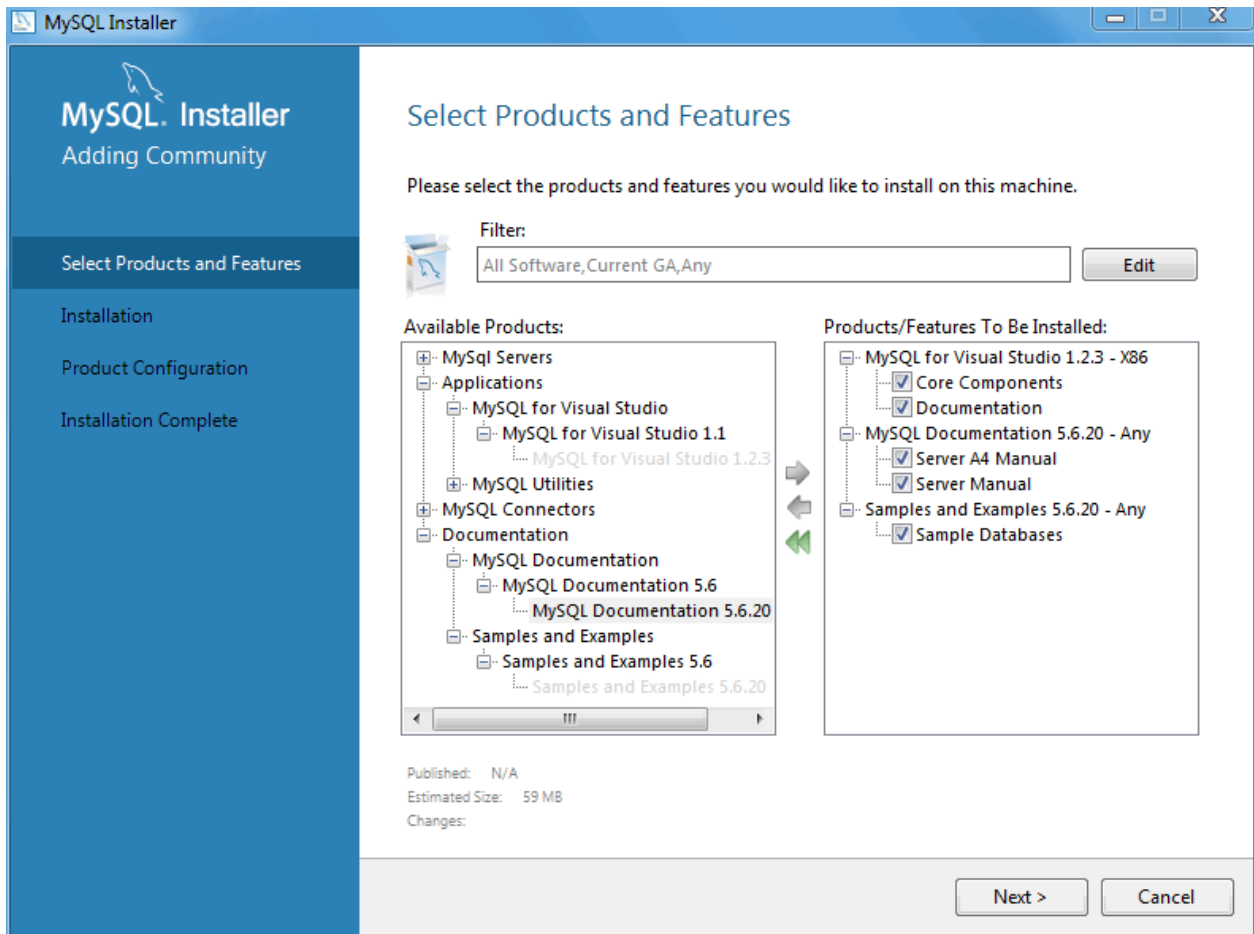
After the MySQL Installer configuration process is finished, MySQL Installer reloads the opening page where you can execute other installation and configuration related actions.

MySQL Installer is added to the Microsoft Windows Start menu under the **MySQL** group. Opening MySQL Installer loads its dashboard where installed MySQL products are listed, and other MySQL Installer actions are available:

Figure 1.12 MySQL Installer - Main Dashboard

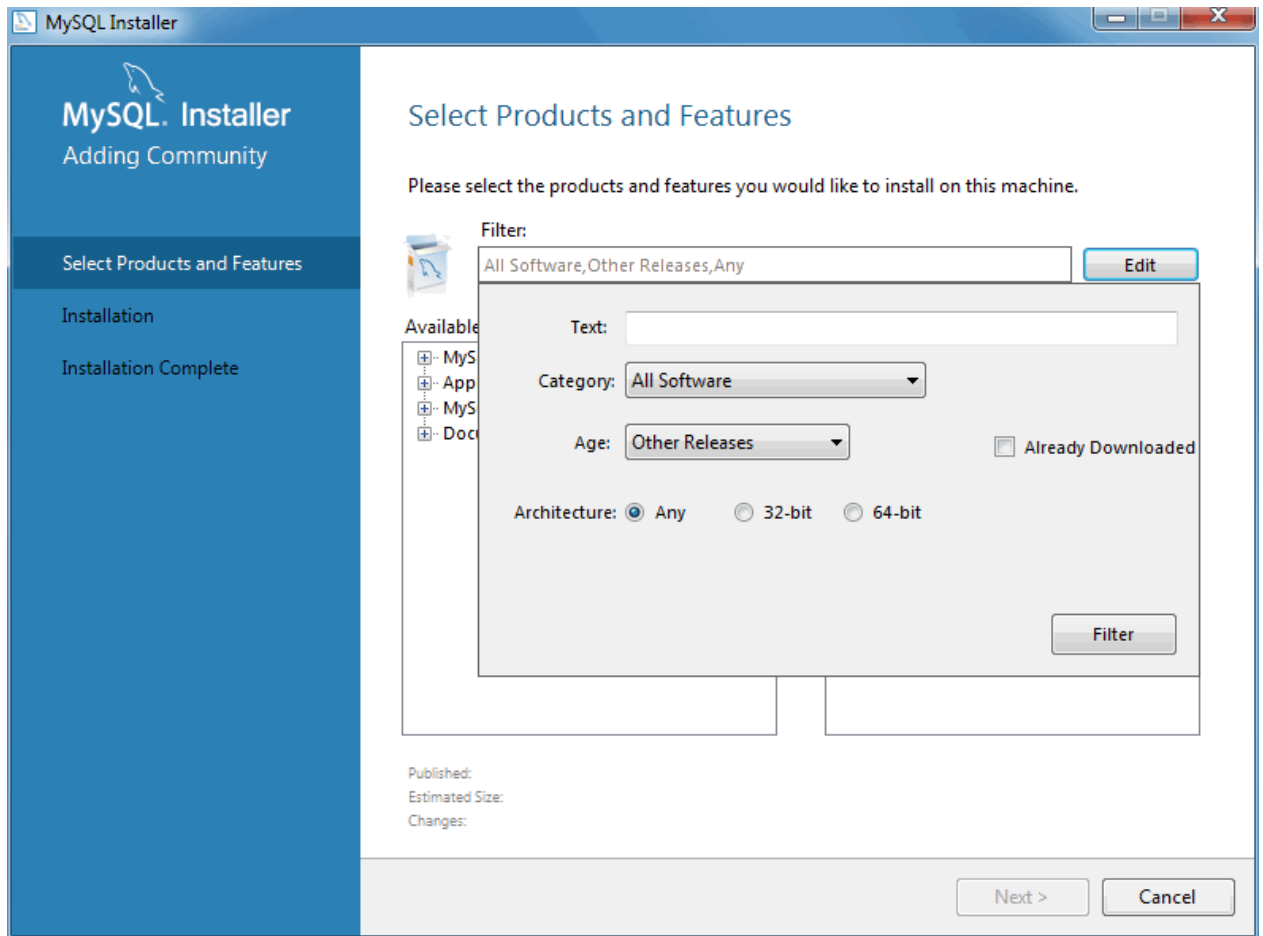
Adding MySQL Products

Click **Add** to add new products. This loads the **Select Products and Features** page:

Figure 1.13 MySQL Installer - Select Products and Features

From here, choose the MySQL products you want to install from the left **Available Products** pane, and then click the green right arrow to queue products for installation.

Optionally, click **Edit** to open the product and features search filter:

Figure 1.14 MySQL Installer - Select Products and Features Filter

For example, you might choose to include Pre-Release products in your selections, such as a Beta product that has not yet reached General Availability (GA) status.

Select all of the MySQL products you want to install, then click **Next** to continue using the defaults, or highlight a selected product and click **Advanced Options** to optionally alter options such as the MySQL server data and installation paths. Click **Execute** to execute the installation process to install all of the selected products.

1.3.1.1 MySQL Product Catalog

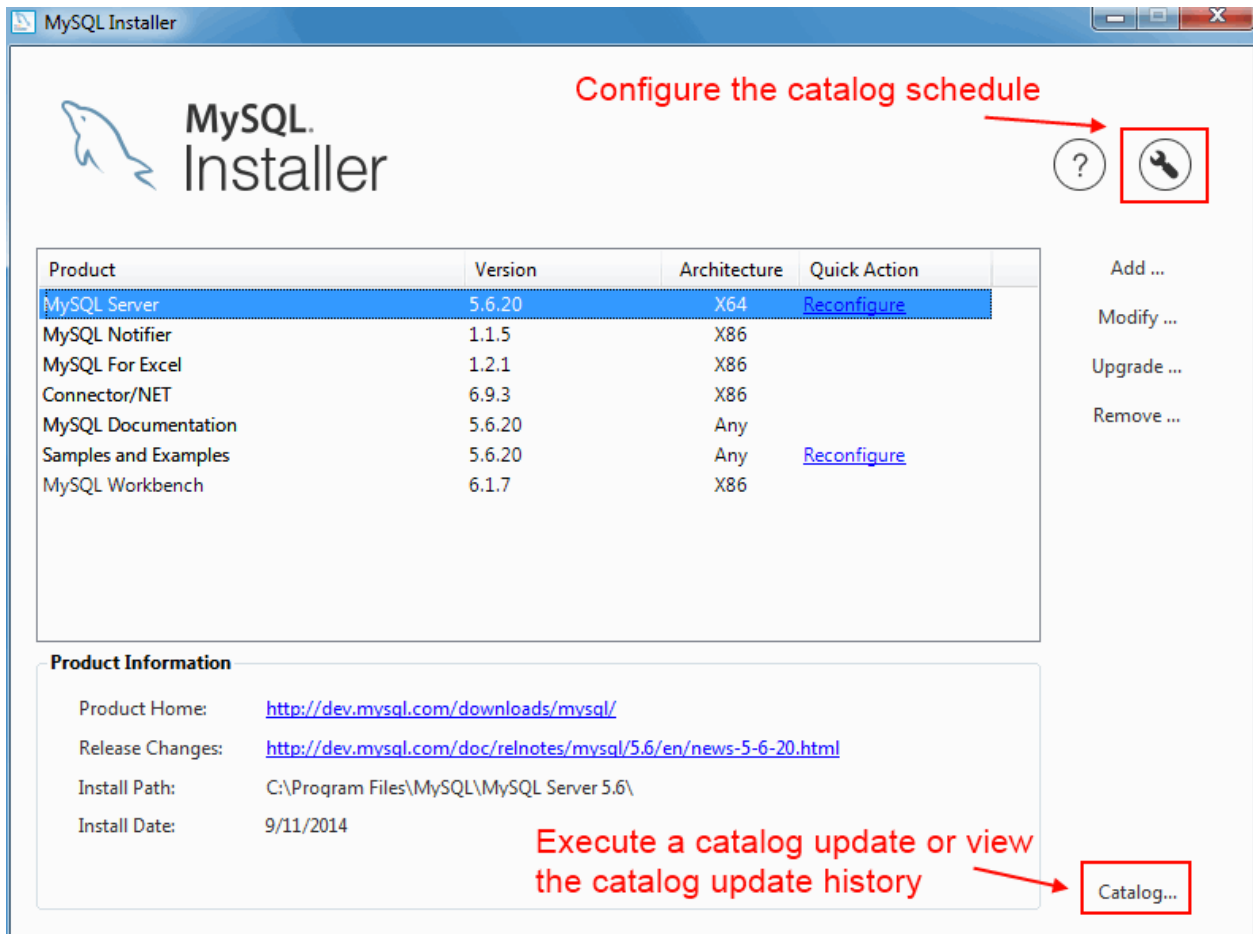
MySQL Installer stores a MySQL product catalog. The catalog can be updated either manually or automatically, and the catalog change history is also available. The automatic update is enabled by default.

Note

The product catalog update also checks for a newer version of MySQL Installer, and prompts for an update if one is present.

Manual updates

You can update the MySQL product catalog at any time by clicking **Catalog** on the Installer dashboard.

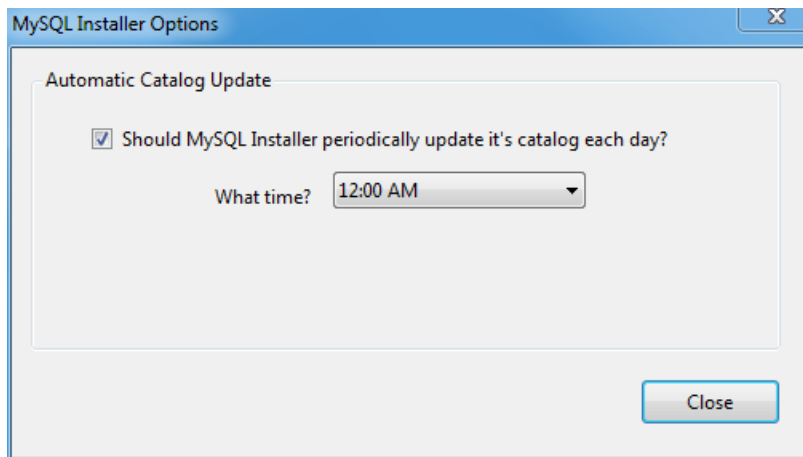
Figure 1.15 MySQL Installer - Open the MySQL Product Catalog

From there, click **Execute** to update the product catalog.

Automatic updates

MySQL Installer can automatically update the MySQL product catalog. By default, this feature is enabled to execute each day at 12:00 AM. To configure this feature, click the wrench icon on the Installer dashboard.

The next window configures the **Automatic Catalog Update**. Enable or disable this feature, and also set the hour.

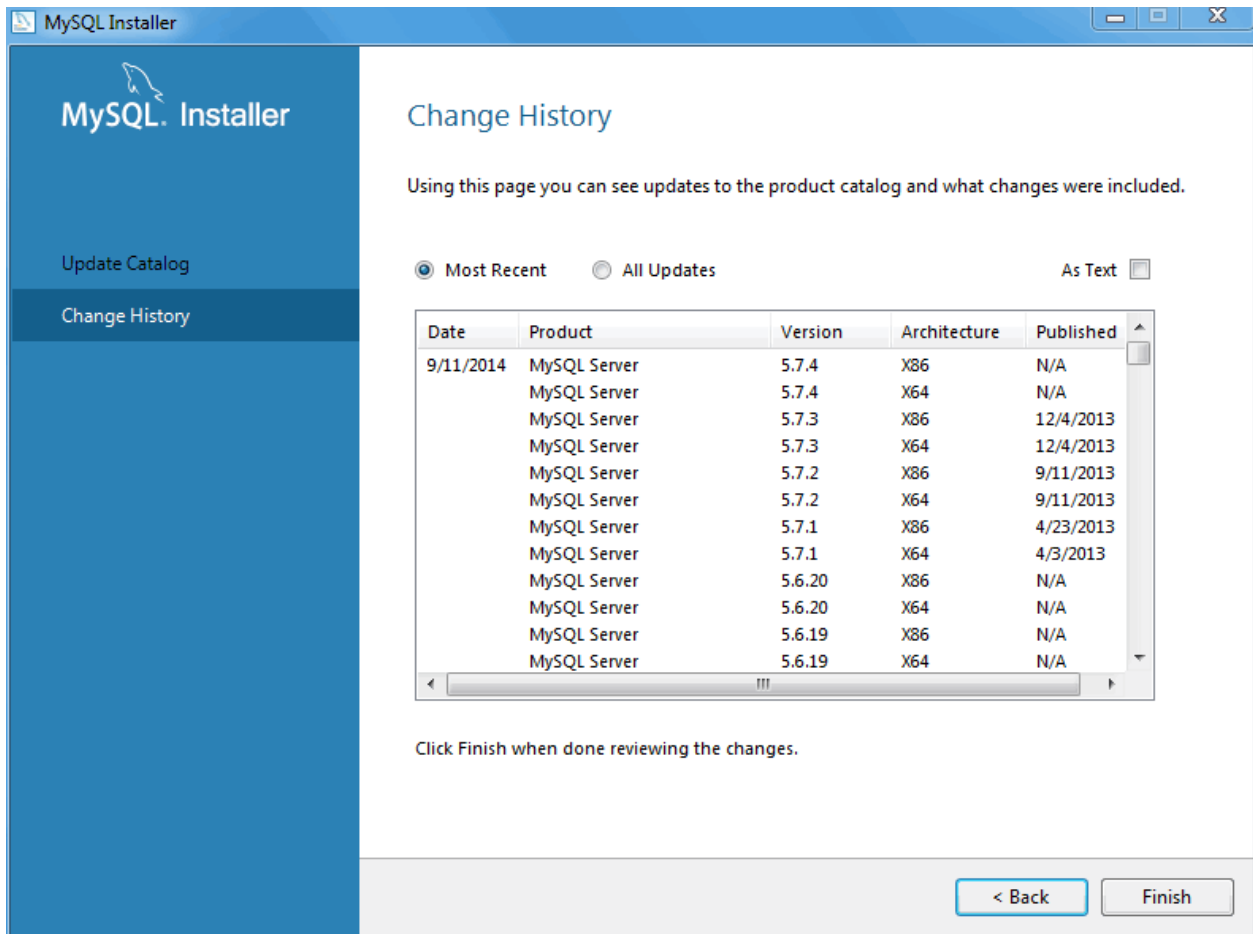
Figure 1.16 MySQL Installer - Configure the Catalog Scheduler

This option uses the Windows Task Scheduler to schedule a task named "ManifestUpdate".

Change History

MySQL Installer tracks the change history for all of the MySQL products. Click **Catalog** from the dashboard, optionally update the catalog (or, toggle the **Do not update at this time** checkbox), click **Next/Execute**, and then view the change history.

Figure 1.17 MySQL Installer - Catalog Change History



1.3.1.2 Remove MySQL Products

MySQL Installer can also remove MySQL products from your system. To remove a MySQL product, click **Remove** from the Installer dashboard. This opens a window with a list of installed MySQL products. Select the MySQL products you want to remove (uninstall), and then click **Execute** to begin the removal process.

Note

To select all MySQL products, click the ☐ checkbox to the left of the **Product** label.

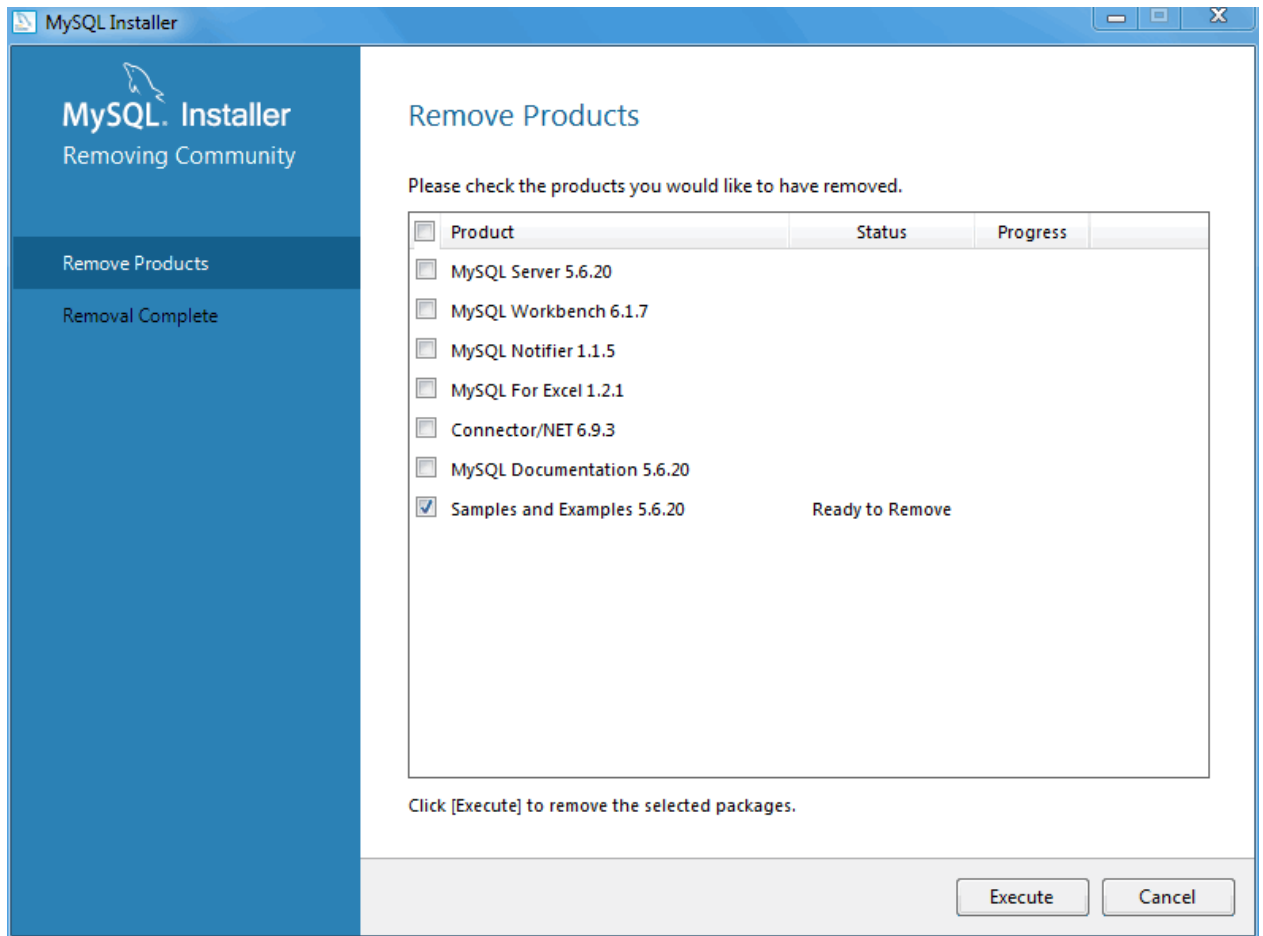
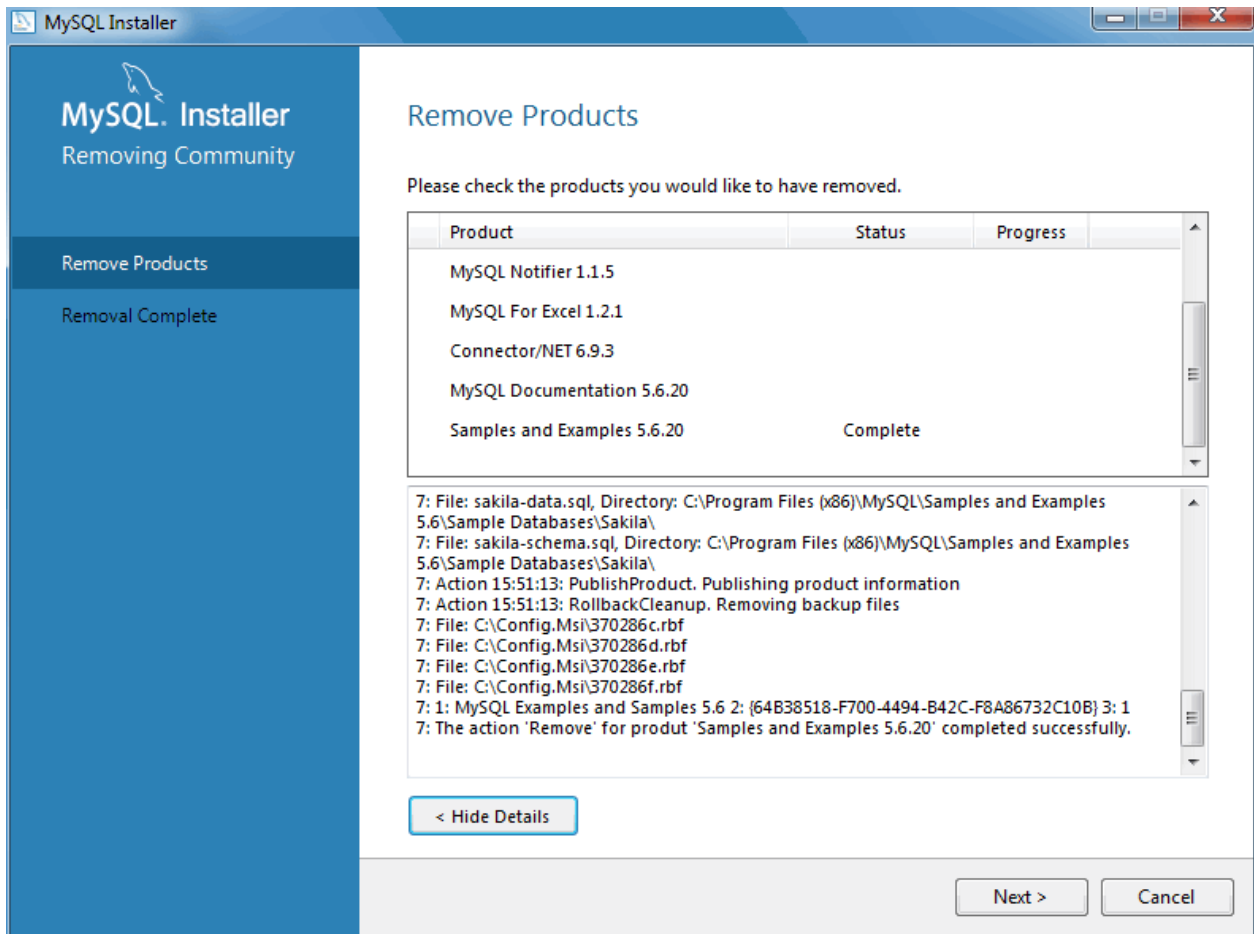
Figure 1.18 MySQL Installer - Removing Products: Select

Figure 1.19 MySQL Installer - Removing Products: Executed

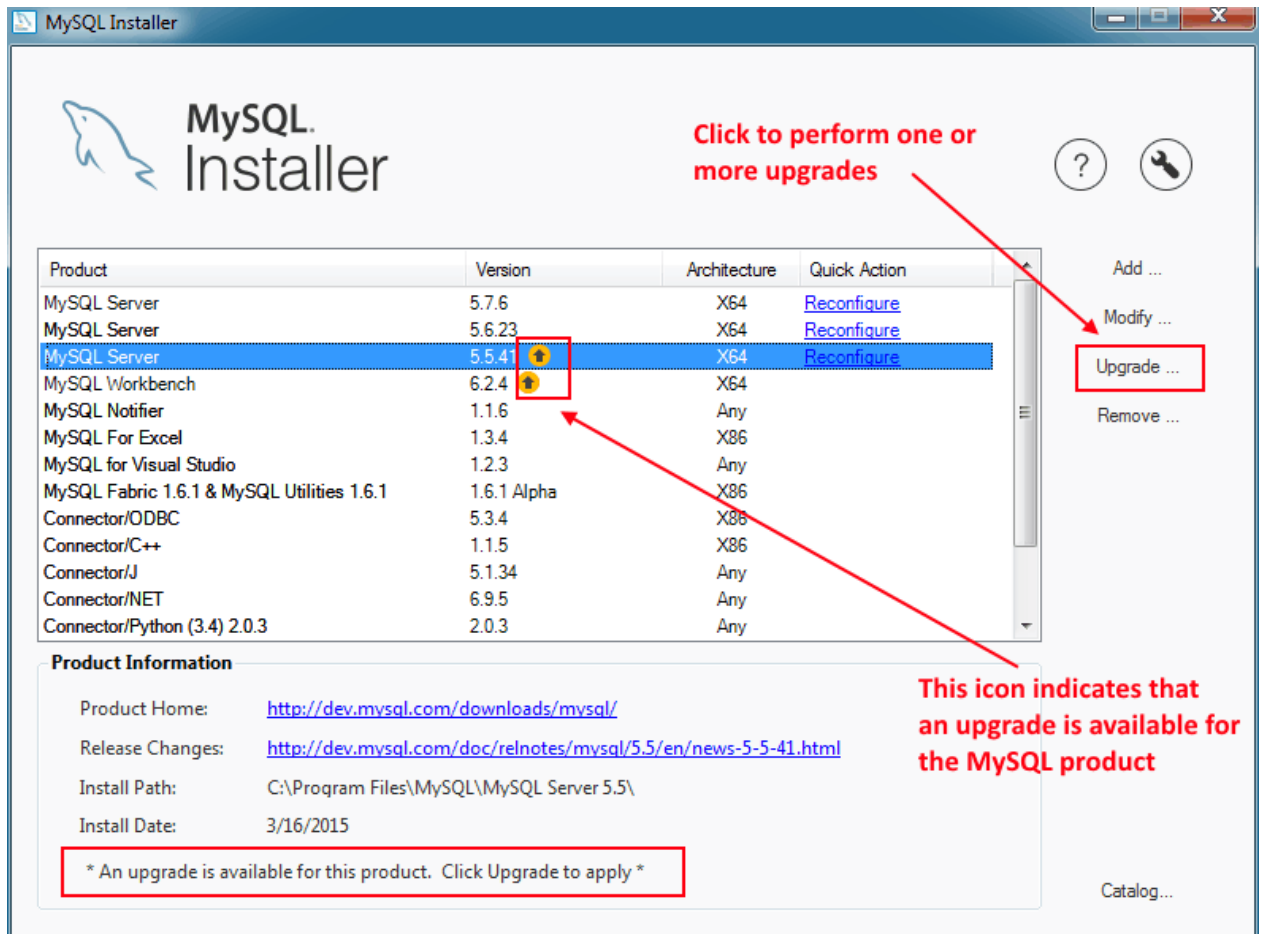
1.3.1.3 Alter MySQL Products

Use MySQL Installer to modify, configure, or upgrade your MySQL product installations.

Upgrade

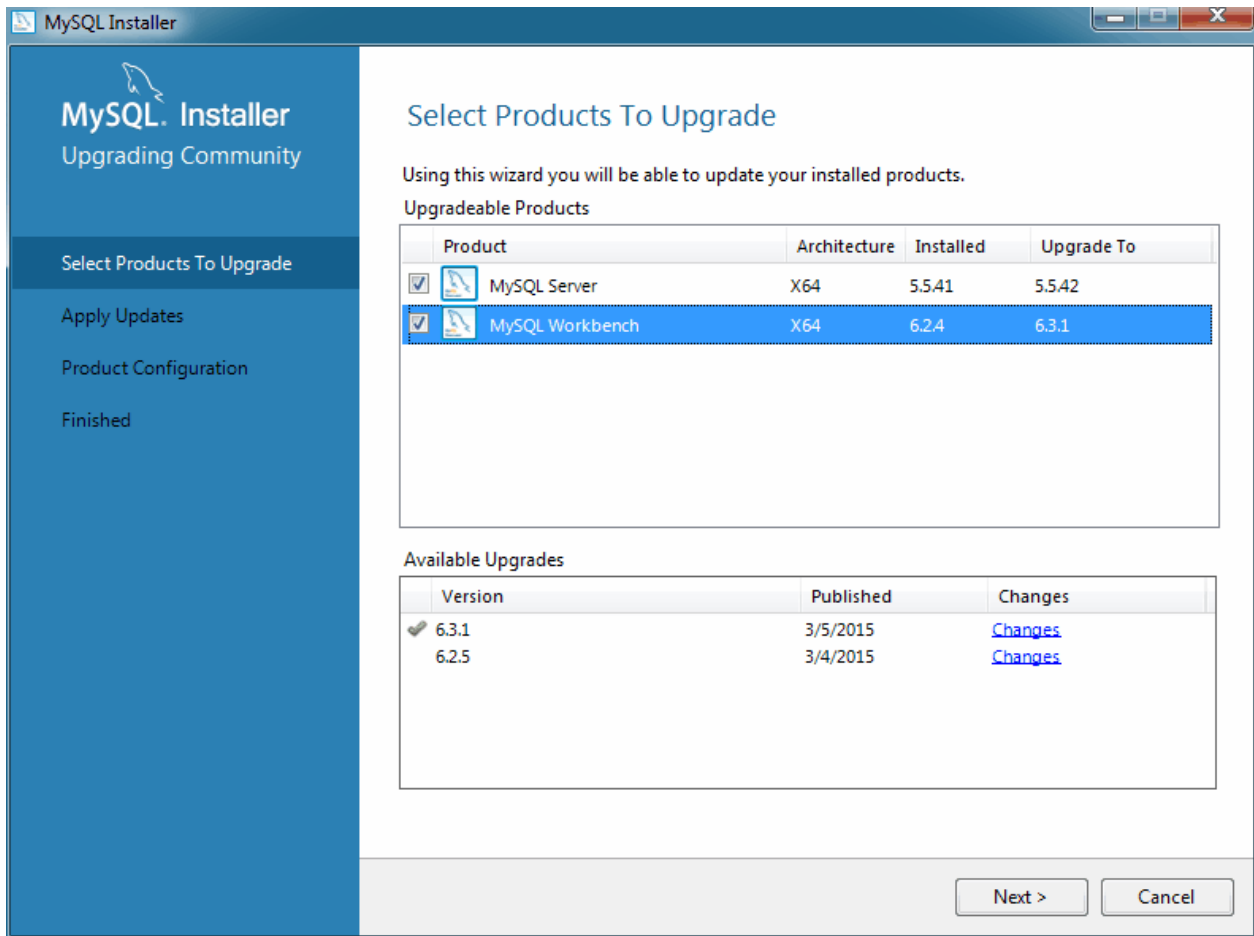
Upgradable MySQL products are listed on the main dashboard with an arrow icon () next to their version number.

Figure 1.20 MySQL Installer - Upgrade a MySQL Product

**Note**

The "upgrade" functionality requires a current product catalog. This catalog is updated either manually or automatically (daily) by enabling the **Automatic Catalog Update** feature. For additional information, see [Section 1.3.1.1, "MySQL Product Catalog"](#).

Click **Upgrade** to upgrade the available products. Our example indicates that MySQL Workbench 6.2.4 can be upgraded version 6.3.1 or 6.2.5, and MySQL server from 5.5.41 to 5.5.42.

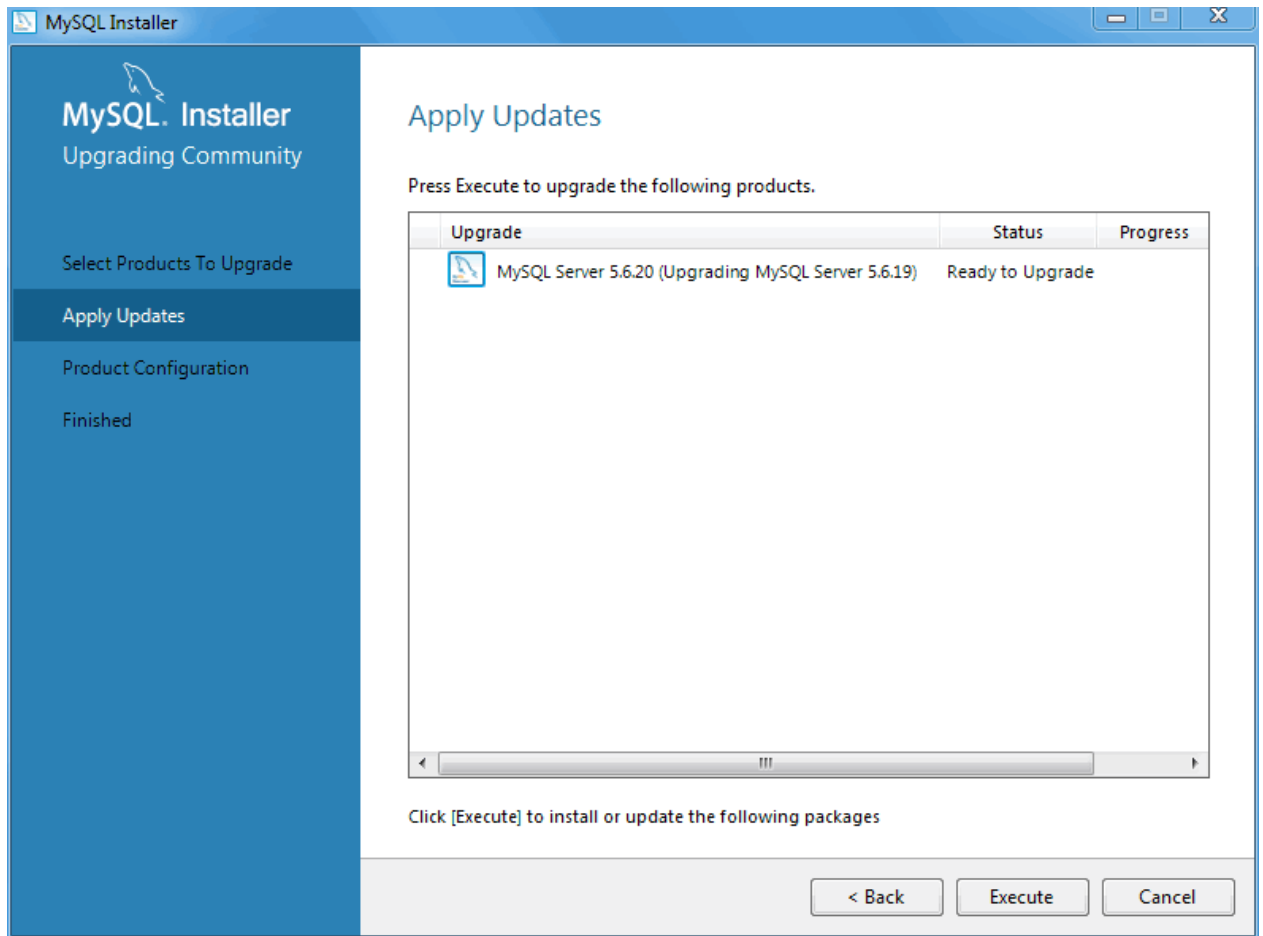
Figure 1.21 MySQL Installer - Select Products To Upgrade

If multiple upgrade versions are available (such as our MySQL Workbench example above), select the desired version for the upgrade in the **Available Upgrades** area.

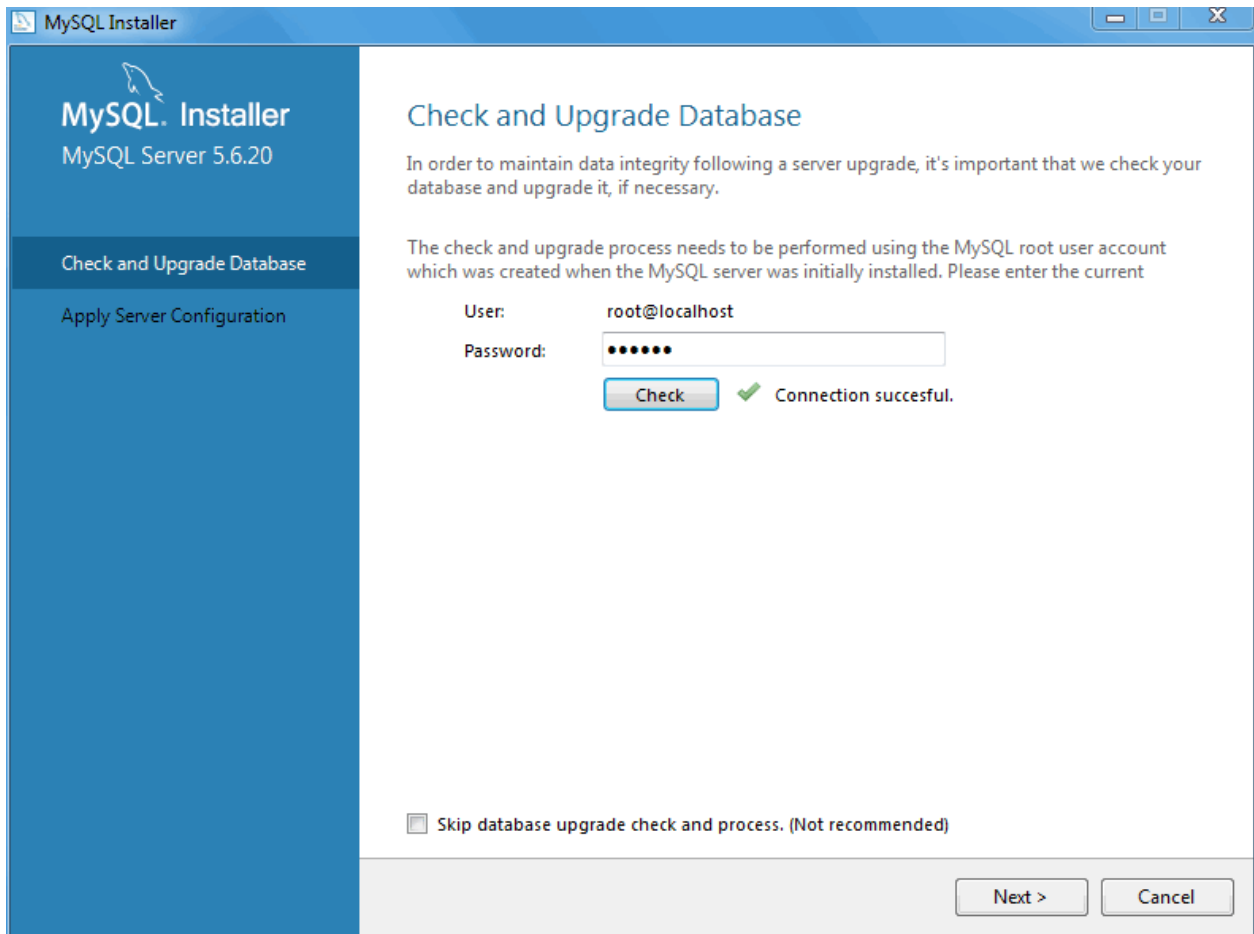
Note

Optionally, click the **Changes** link to view the version's release notes.

After selecting (checking) the products and versions to upgrade, click **Next** to begin the upgrade process.

Figure 1.22 MySQL Installer - Apply Updates

A MySQL server upgrade will also check and upgrade the server's database. Although optional, this step is recommended.

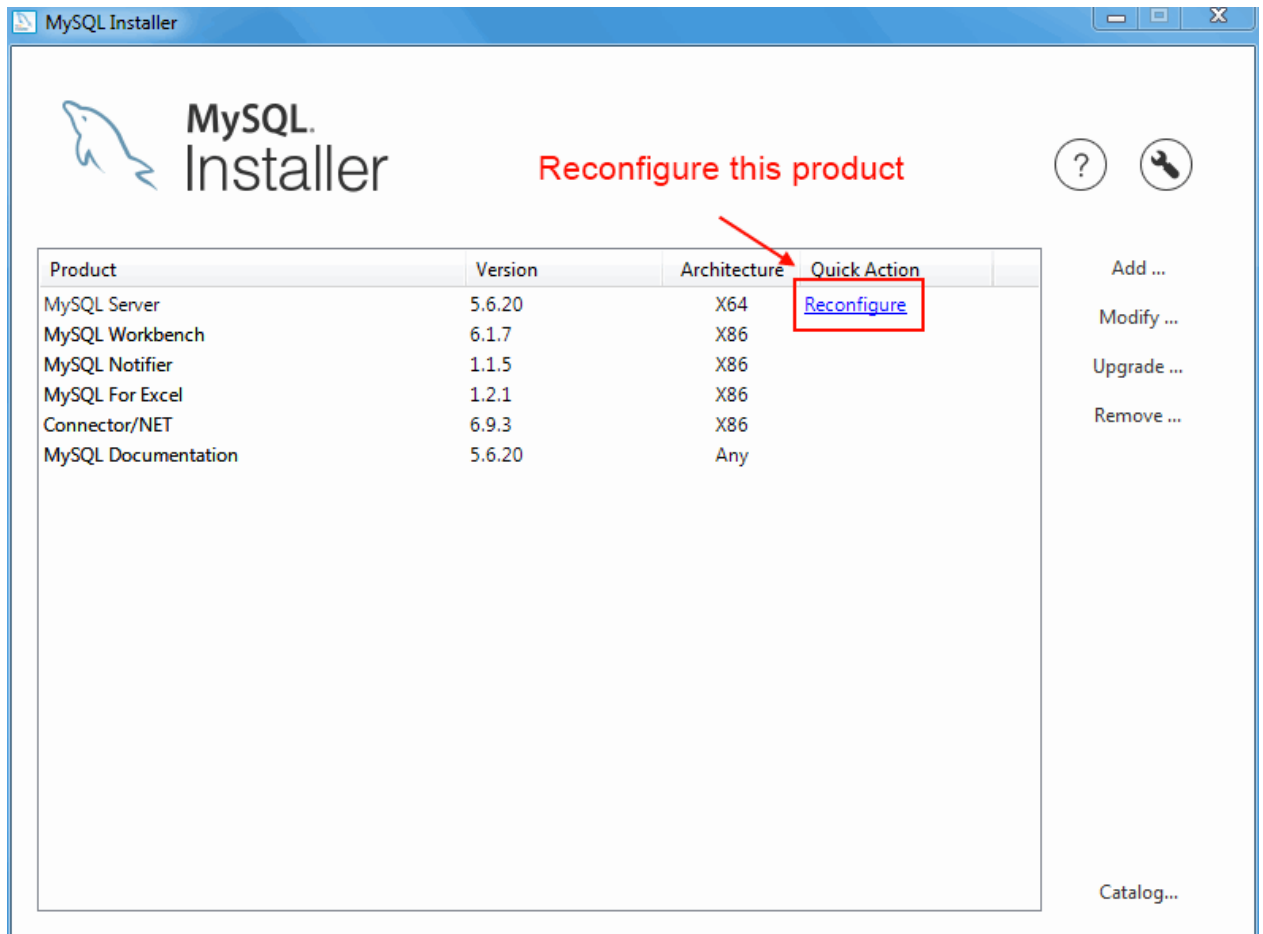
Figure 1.23 MySQL Installer - Check and Upgrade Database

Upon completion, your upgraded products will be upgraded and available to use. A MySQL server upgrade also restarts the MySQL server.

Reconfigure

Some MySQL products, such as the MySQL server, include a **Reconfigure** option. It opens the same configuration options that were set when the MySQL product was installed, and is pre-populated with the current values.

To execute, click the [Reconfigure](#) link under the **Quick Action** column on the main dashboard for the MySQL product that you want to reconfigure.

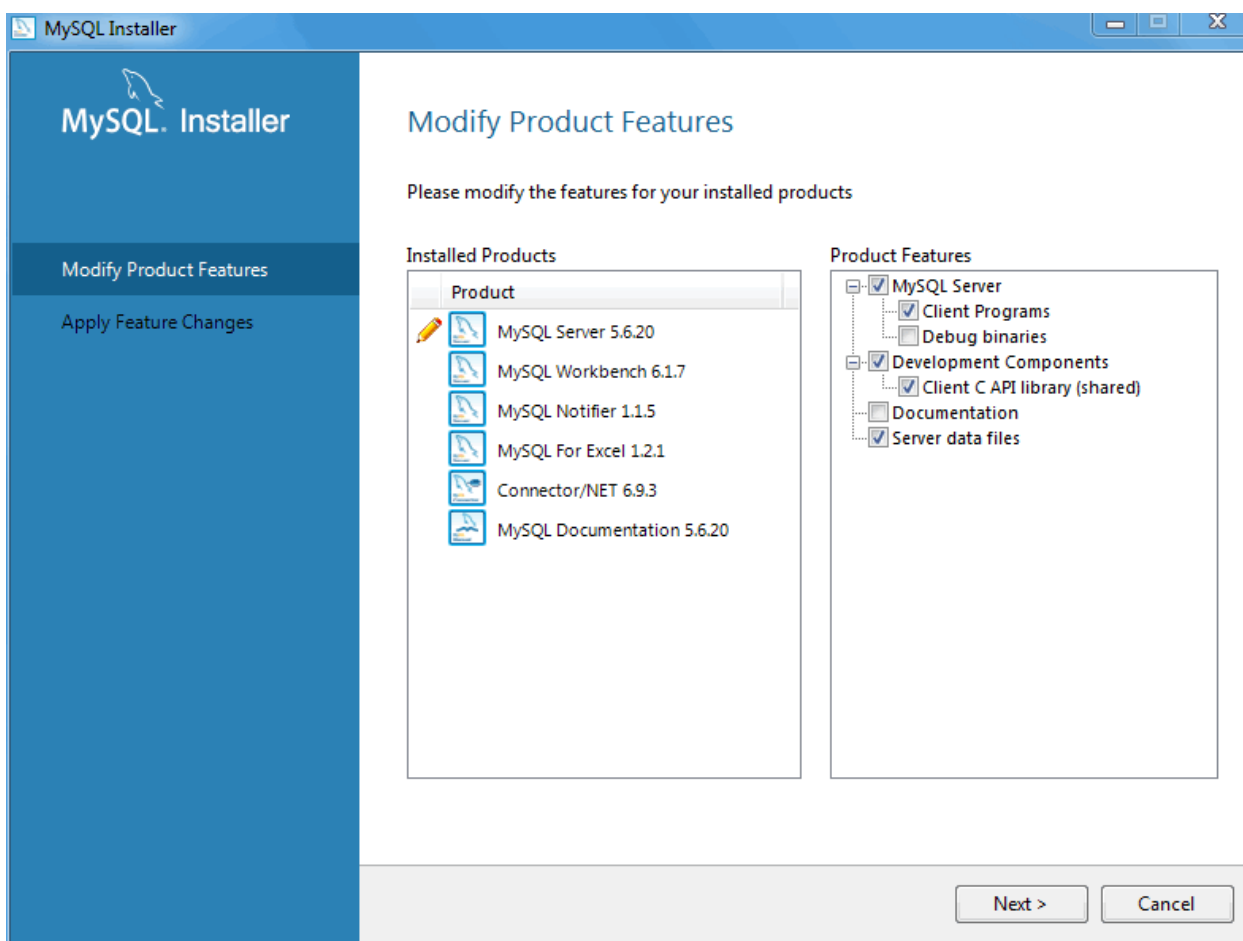
Figure 1.24 MySQL Installer - Reconfigure a MySQL Product

In the case of the MySQL server, this opens a configuration wizard that relates to the selected product. For example, for MySQL Server this includes setting the type, ports, log paths, and so on.

Modify

Many MySQL products contain feature components that can be added or removed. For example, [Debug binaries](#) and [Client Programs](#) are subcomponents of the MySQL server.

To modify the features of a product, click **Modify** on the main dashboard.

Figure 1.25 MySQL Installer - Modify Product Features

Click **Execute** to execute the modification request.

1.3.2 MySQL Installer Console

`MySQLInstallerConsole` provides functionality similar to the GUI version of MySQL Installer, but from the command-line. It is installed when MySQL Installer is initially executed, and then available within the `MySQL Installer` directory. Typically that is in `C:\Program Files (x86)\MySQL\MySQL Installer\`, and the console must be executed with administrative privileges.

To use, invoke the Command Prompt with administrative privileges by choosing **Start, Accessories**, then right-click on **Command Prompt** and choose **Run as administrator**. And from the command-line, optionally change the directory to where `MySQLInstallerConsole` is located:

```
C:\> cd "C:\Program Files (x86)\MySQL\MySQL Installer for Windows"
C:\> MySQLInstallerConsole.exe help

C:\Program Files (x86)\MySQL\MySQL Installer for Windows>MySQLInstallerConsole.exe help

The following commands are available:

Configure - Configures one or more of your installed programs.
Help      - Provides list of available commands.
Install   - Install and configure one or more available MySQL programs.
List      - Provides an interactive way to list all products available.
Modify    - Modifies the features of installed products.
```

Remove	- Removes one or more products from your system.
Status	- Shows the status of all installed products.
Update	- Update the current product catalog.
Upgrade	- Upgrades one or more of your installed programs.

MySQLInstallerConsole supports the following options, which are specified on the command line:

Note

Configuration block values that contain a colon (":") must be wrapped in double quotes. For example, `installdir="C:\MySQL\MySQL Server 5.6"`.

- **configure** [product1]:[setting]=[value]; [product2]:[setting]=[value]; [...]

Configure one or more MySQL products on your system. Multiple setting=value pairs can be configured for each product.

Switches include:

- **-showsettings** : Displays the available options for the selected product, by passing in the product name after **-showsettings**.
- **-silent** : Disable confirmation prompts.

```
C:\> MySQLInstallerConsole configure -showsettings server
C:\> MySQLInstallerConsole configure server:port=3307
```

- **help** [command]

Displays a help message with usage examples, and then exits. Pass in an additional command to receive help specific to that command.

```
C:\> MySQLInstallerConsole help
C:\> MySQLInstallerConsole help install
```

- **install** [product]:[features]:[config block]:[config block]:[config block]; [...]

Install one or more MySQL products on your system.

Switches and syntax options include:

- **-type=[SetupType]** : Installs a predefined set of software. The "SetupType" can be one of the following:

Note

Non-custom setup types can only be chosen if no other MySQL products are installed.

- **Developer**: Installs a complete development environment.
- **Server**: Installs a single MySQL server
- **Client**: Installs client programs and libraries
- **Full**: Installs everything
- **Custom**: Installs user selected products. This is the default option.

- `-showsettings` : Displays the available options for the selected product, by passing in the product name after `-showsettings`.
- `-silent` : Disable confirmation prompts.
- `[config block]`: One or more configuration blocks can be specified. Each configuration block is a semicolon separated list of key value pairs. A block can include either a "config" or "user" type key, where "config" is the default type if one is not defined.

Configuration block values that contain a colon (":") must be wrapped in double quotes. For example, `installdir="C:\MySQL\MySQL Server 5.6"`.

Only one "config" type block can be defined per product. A "user" block should be defined for each user that should be created during the product's installation.

Note

Adding users is not supported when a product is being reconfigured.

- `[feature]`: The feature block is a semicolon separated list of features, or '*' to select all features.

```
C:\> MySQLInstallerConsole install server;5.6.25:*:port=3307;serverid=2:type=user;username=foo;password=bar;
C:\> MySQLInstallerConsole install server;5.6.25;x64 -silent
```

An example that passes in additional configuration blocks, broken up by ^ to fit this screen:

```
C:\> MySQLInstallerConsole install server;5.6.25;x64:*:type=config;openfirewall=true; ^
      generallog=true;binlog=true;serverid=3306;enable_tcpip=true;port=3306;rootpasswd=pass; ^
      installdir="C:\MySQL\MySQL Server 5.6":type=user;datadir="C:\MySQL\data";username=foo;password=bar
```

- `list`

Lists an interactive console where all of the available MySQL products can be searched. Execute `MySQLInstallerConsole list` to launch the console, and enter in a substring to search.

```
C:\> MySQLInstallerConsole list
```

- `modify [product1:-removelist|+addlist] [product2:-removelist|+addlist] [...]`

Modifies or displays features of a previously installed MySQL product.

- `-silent` : Disable confirmation prompts.

```
C:\> MySQLInstallerConsole modify server
C:\> MySQLInstallerConsole modify server:+documentation
C:\> MySQLInstallerConsole modify server:-debug
```

- `remove [product1] [product2] [...]`

Removes one or more products from your system.

- `*` : Pass in `*` to remove all of the MySQL products.
- `-continue` : Continue the operation even if an error occurs.

- `-silent` : Disable confirmation prompts.

```
C:\> MySQLInstallerConsole remove *
C:\> MySQLInstallerConsole remove server
```

- `status`

Provides a quick overview of the MySQL products that are installed on the system. Information includes product name and version, architecture, date installed, and install location.

```
C:\> MySQLInstallerConsole status
```

- `upgrade [product1:version] [product2:version], [...]`

Upgrades one or more products on your system. Syntax options include:

- `*` : Pass in `*` to upgrade all products to the latest version, or pass in specific products.
- `!` : Pass in `!` as a version number to upgrade the MySQL product to its latest version.
- `-silent` : Disable confirmation prompts.

```
C:\> MySQLInstallerConsole upgrade *
C:\> MySQLInstallerConsole upgrade workbench:6.3.5
C:\> MySQLInstallerConsole upgrade workbench:!
C:\> MySQLInstallerConsole upgrade workbench:6.3.5 excel:1.3.2
```

- `update`

Downloads the latest MySQL product catalog to your system. On success, the download catalog will be applied the next time either MySQL Installer or MySQLInstallerConsole is executed.

```
C:\> MySQLInstallerConsole update
```

Note

The **Automatic Catalog Update** GUI option executes this command from the Windows Task Scheduler.

1.4 MySQL Notifier

MySQL Notifier is a tool that enables you to monitor and adjust the status of your local and remote MySQL server instances through an indicator that resides in the system tray. MySQL Notifier also gives quick access to MySQL Workbench through its context menu.

The MySQL Notifier is installed by MySQL Installer, and (by default) will start-up when Microsoft Windows is started.

To install, download and execute the [MySQL Installer](#), be sure the MySQL Notifier product is selected, then proceed with the installation. See the [MySQL Installer manual](#) for additional details.

For notes detailing the changes in each release of MySQL Notifier, see the [MySQL Notifier Release Notes](#).

Visit the [MySQL Notifier forum](#) for additional MySQL Notifier help and support.

Features include:

- Start, Stop, and Restart instances of the MySQL Server.
- Automatically detects (and adds) new MySQL Server services. These are listed under **Manage Monitored Items**, and may also be configured.
- The Tray icon changes, depending on the status. It's green if all monitored MySQL Server instances are running, or red if at least one service is stopped. The **Update MySQL Notifier tray icon based on service status** option, which dictates this behavior, is enabled by default for each service.
- Links to other applications like MySQL Workbench, MySQL Installer, and the MySQL Utilities. For example, choosing **Manage Instance** will load the MySQL Workbench Server Administration window for that particular instance.
- If MySQL Workbench is also installed, then the **Manage Instance** and **SQL Editor** options are available for local (but not remote) MySQL instances.
- Monitors both local and remote MySQL instances.

1.4.1 MySQL Notifier Usage

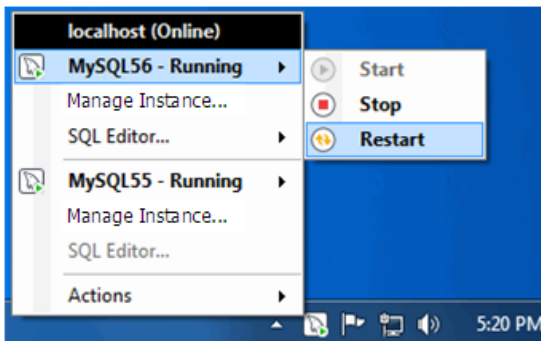
MySQL Notifier resides in the system tray and provides visual status information for your MySQL server instances. A green icon is displayed at the top left corner of the tray icon if the current MySQL server is running, or a red icon if the service is stopped.

MySQL Notifier automatically adds discovered MySQL services on the local machine, and each service is saved and configurable. By default, the **Automatically add new services whose name contains** option is enabled and set to `mysql`. Related **Notifications Options** include being notified when new services are either discovered or experience status changes, and are also enabled by default. And uninstalling a service will also remove the service from MySQL Notifier.

Clicking the system tray icon will reveal several options, as the follow figures show:

The Service Instance menu is the main MySQL Notifier window, and enables you to Stop, Start, and Restart the MySQL server.

Figure 1.26 MySQL Notifier Service Instance menu

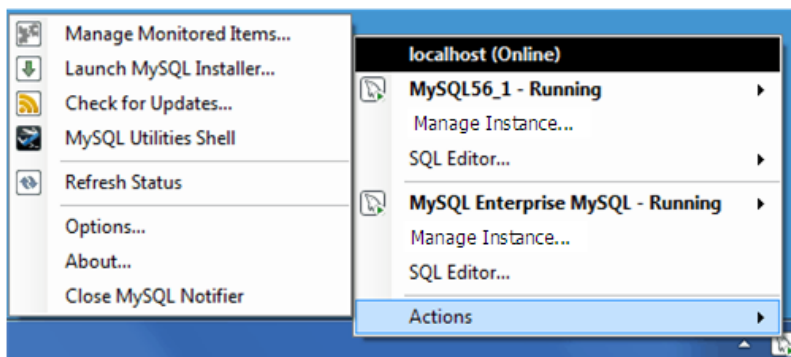


The **Actions** menu includes several links to external applications (if they are installed), and a **Refresh Status** option to manually refresh the status of all monitored services (in both local and remote computers) and MySQL instances.

Note

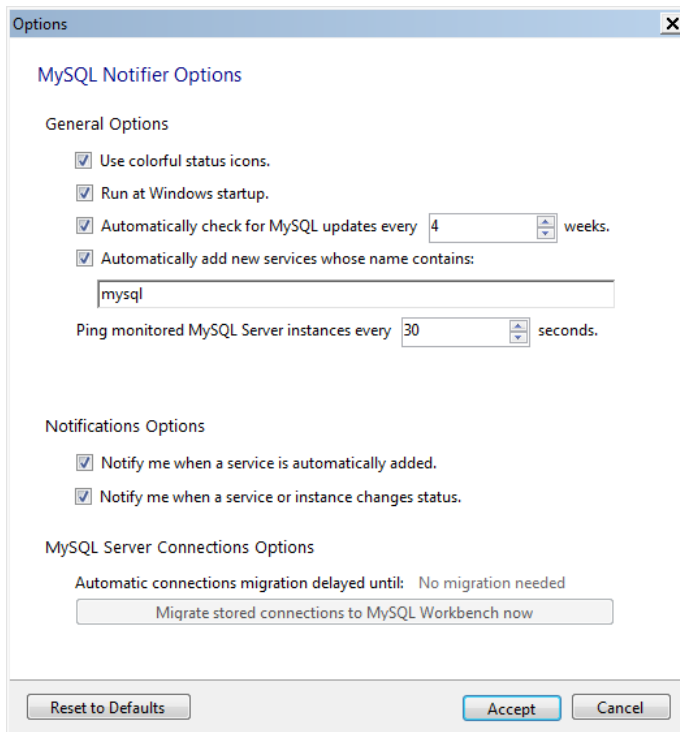
The main menu will not show the **Actions** menu when there are no services being monitored by MySQL Notifier.

Figure 1.27 MySQL Notifier Actions menu

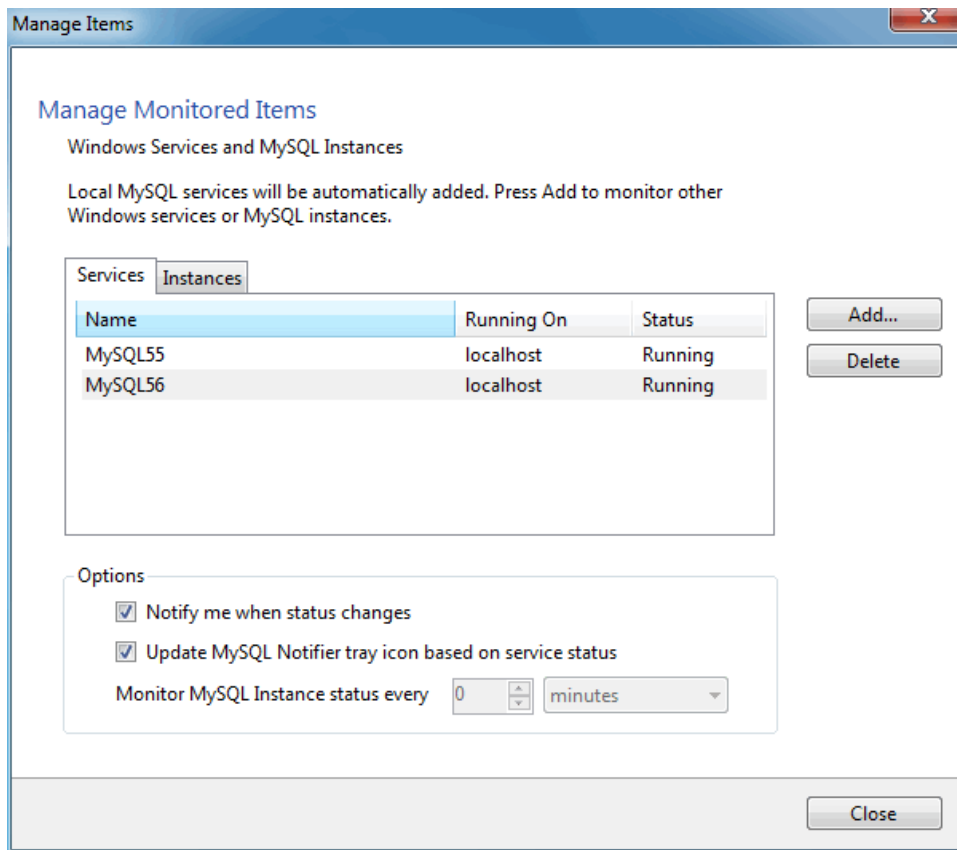


The **Actions**, **Options** menu configures MySQL Notifier and includes options to:

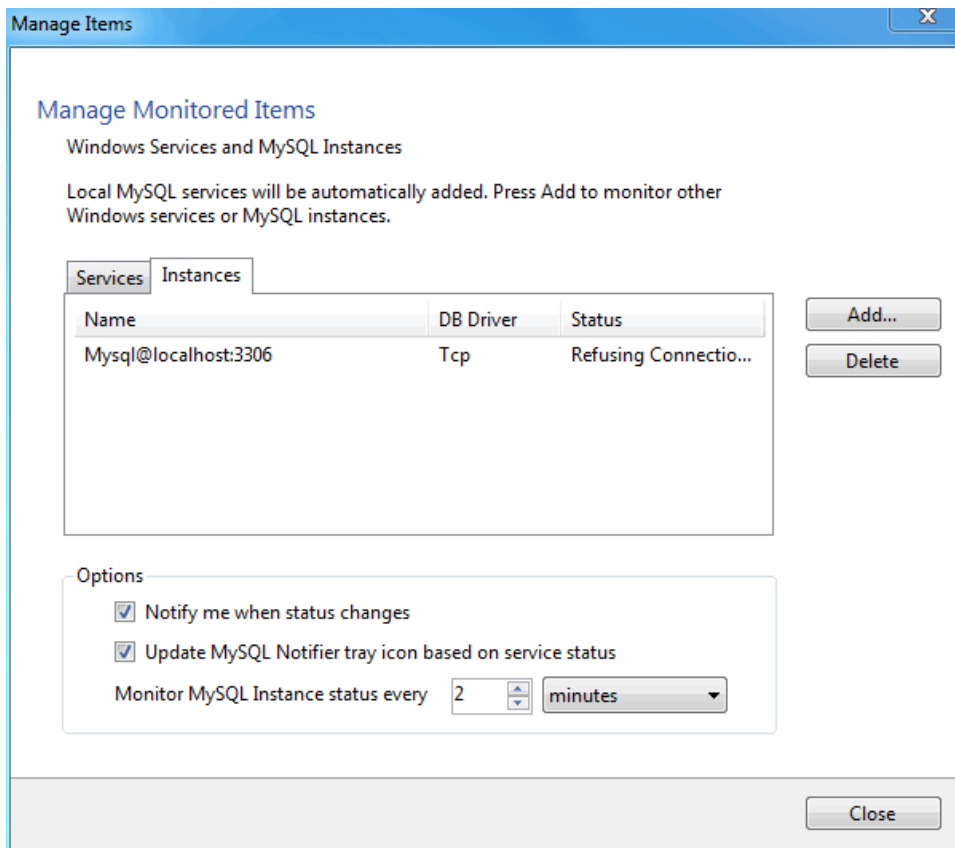
- **Use colorful status icons:** Enables a colorful style of icons for the tray of MySQL Notifier.
- **Run at Windows Startup:** Allows the application to be loaded when Microsoft Windows starts.
- **Automatically Check For Updates Every # Weeks:** Checks for a new version of MySQL Notifier, and runs this check every # weeks.
- **Automatically add new services whose name contains:** The text used to filter services and add them automatically to the monitored list of the local computer running MySQL Notifier, and on remote computers already monitoring Windows services.
- **Ping monitored MySQL Server instances every # seconds:** The interval (in seconds) to ping monitored MySQL Server instances for status changes. Longer intervals might be necessary if the list of monitored remote instances is large.
- **Notify me when a service is automatically added:** Will display a balloon notification from the taskbar when a newly discovered service is added to the monitored services list.
- **Notify me when a service changes status:** Will display a balloon notification from the taskbar when a monitored service changes its status.
- **Automatic connections migration delayed until:** When there are connections to migrate, postpone the migration by one hour, one day, one week, one month, or indefinitely.

Figure 1.28 MySQL Notifier Options menu

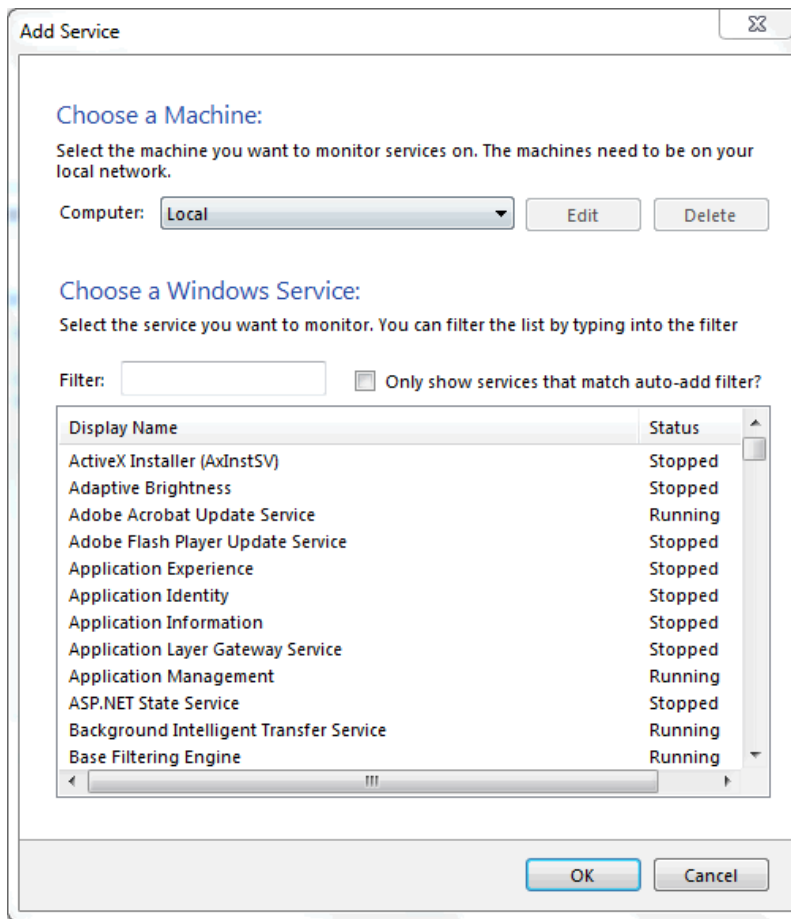
The **Actions**, **Manage Monitored Items** menu enables you to configure the monitored services and MySQL instances. First, with the **Services** tab open:

Figure 1.29 MySQL Notifier Manage Services menu

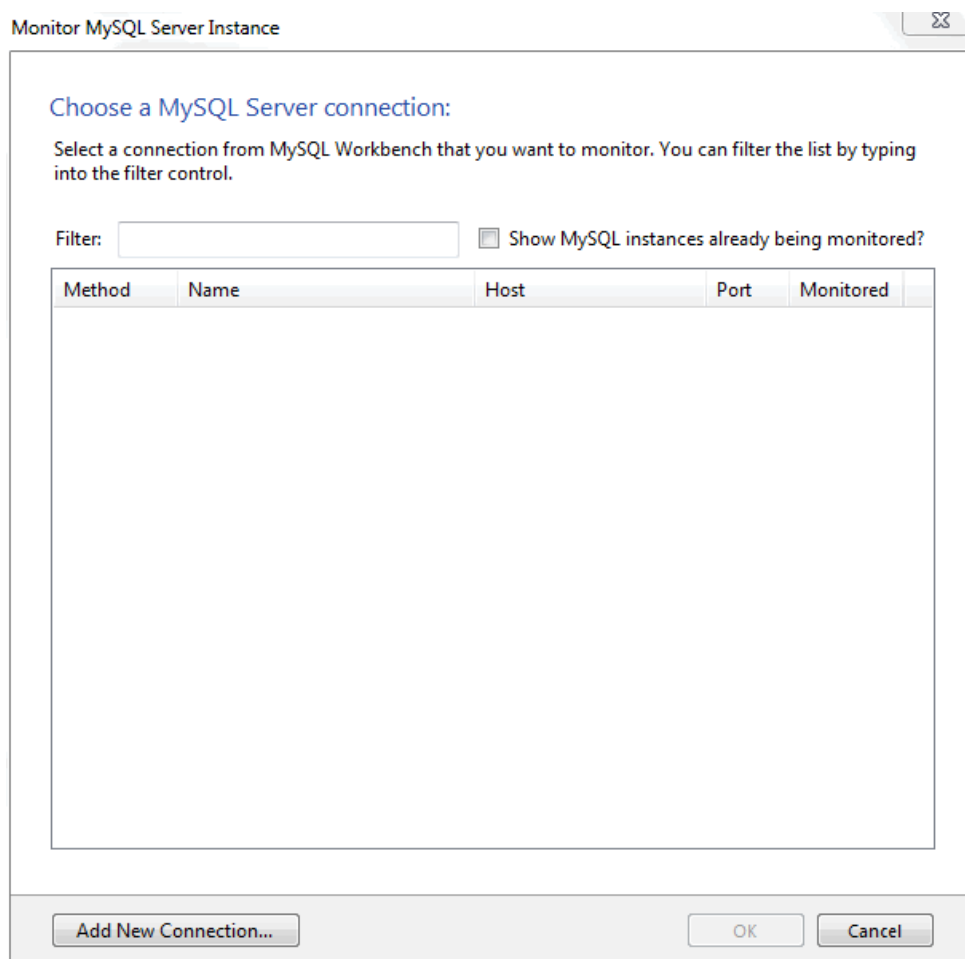
The **Instances** tab is similar:

Figure 1.30 MySQL Notifier Manage Instances menu

Adding a service or instance (after clicking **Add** in the **Manage Monitored Items** window) enables you to select a running Microsoft Windows service or instance connection, and configure MySQL Notifier to monitor it. Add a new service or instance by clicking service name from the list, then **OK** to accept. Multiple services and instances may be selected.

Figure 1.31 MySQL Notifier Adding new services

Add instances:

Figure 1.32 MySQL Notifier Adding new instances

Troubleshooting

For issues that are not documented here, visit the [MySQL Notifier Support Forum](#) for MySQL Notifier help and support.

- *Problem:* attempting to start/stop/restart a MySQL service might generate an error similar to "The Service **MySQLVERSION** failed the most recent status change request with the message "The service **mysqlVERSION** was not found in the Windows Services".

Explanation: this is a case-sensitivity issue, in that the service name is **MySQLVERSION** compared to having **mysqlVERSION** in the configuration file.

Solution: either update your MySQL Notifier configuration file with the correct information, or stop MySQL Notifier and delete this configuration file. The MySQL Notifier configuration file is located at `%APPDATA%\Oracle\MySQL Notifier\settings.config` where `%APPDATA%` is a variable and depends on your system. A typical location is "C:\Users\YourUsername\AppData\Running\Oracle\MySQL Notifier\settings.config" where *YourUsername* is your system's user name. In this file, and within the `ServerList` section, change the `ServerName` values from lowercase to the actual service names. For example, change **mysqlVERSION** to **MySQLVERSION**, save, and then restart MySQL Notifier. Alternatively, stop MySQL Notifier, delete this file, then restart MySQL Notifier.

- *Problem:* when connecting to a remote computer for the purpose of monitoring a remote Windows service, the **Add Service** dialog does not always show all the services shown in the Windows Services console.

Explanation: this behavior is governed by the operating system and the outcome is expected when working with nondomain user accounts. For a complete description of the behavior, see the [User Account Control and WMI](#) article from Microsoft.

Solution: when the remote computer is in a compatible domain, it is recommended that domain user accounts are used to connect through WMI to a remote computer. For detailed setup instructions using WMI, see [Section 1.4.2, “Setting Up Remote Monitoring in MySQL Notifier”](#).

Alternatively, when domain user accounts are not available, Microsoft provides a less secure workaround that should only be implemented with caution. For more information, see the [Description of User Account Control and remote restrictions in Windows Vista](#) KB article from Microsoft.

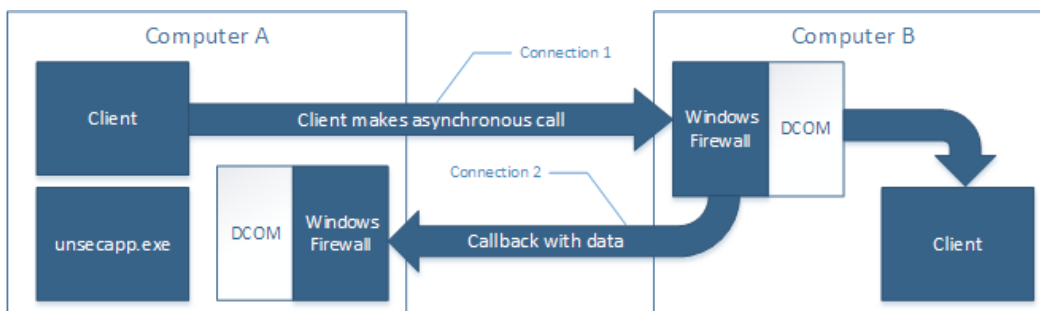
1.4.2 Setting Up Remote Monitoring in MySQL Notifier

MySQL Notifier uses Windows Management Instrumentation (WMI) to manage and monitor services on remote computers. This section explains how it works and how to set up your system to monitor remote MySQL instances.

In order to configure WMI, it is important to understand that the underlying Distributed Component Object Model (DCOM) architecture is doing the WMI work. Specifically, MySQL Notifier is using asynchronous notification queries on remote Microsoft Windows hosts as .NET events. These events send an asynchronous callback to the computer running MySQL Notifier so it knows when a service status has changed on the remote computer. Asynchronous notifications offer the best performance compared to semisynchronous notifications or synchronous notifications that use timers.

Asynchronous notification requires the remote computer to send a callback to the client computer (thus opening a reverse connection), so the Windows Firewall and DCOM settings must be properly configured for the communication to function properly.

Figure 1.33 MySQL Notifier Distributed Component Object Model (DCOM)



Most of the common errors thrown by asynchronous WMI notifications are related to Windows Firewall blocking the communication, or to DCOM / WMI settings not being set up properly. For a list of common errors with solutions, see [Common Errors](#).

The following steps are required to make WMI function. These steps are divided between two machines. A single host computer that runs MySQL Notifier (Computer A), and multiple remote machines that are being monitored (Computer B).

Computer running MySQL Notifier (Computer A)

1. Enable remote administration by either editing the **Group Policy Editor**, or using **NETSH**:

Using the **Group Policy Editor**:

- a. Click **Start**, click **Run**, type `GPEDIT.MSC`, and then click **OK**.
- b. Under the **Local Computer Policy** heading, expand **Computer Configuration**.
- c. Expand **Administrative Templates**, then **Network**, **Network Connections**, and then **Windows Firewall**.
- d. If the computer is in the domain, then double-click **Domain Profile**; otherwise, double-click **Standard Profile**.
- e. Double-click **Windows Firewall: Allow inbound remote administration exception** to open a configuration window.
- f. Check the **Enabled** option button and then click **OK**.

Using the `NETSH` command:

Note

The "netsh firewall" command is deprecated as of Microsoft Server 2008 and Vista, and replaced with "netsh advfirewall firewall".

- a. Open a command prompt window with Administrative rights (you can right-click the Command Prompt icon and select **Run as Administrator**).
- b. Execute the following command:

```
NETSH advfirewall firewall set service RemoteAdmin enable
```

2. Open the DCOM port TCP 135:

- a. Open a command prompt window with Administrative rights (you can right-click the Command Prompt icon and select **Run as Administrator**).
- b. Execute the following command:

```
NETSH advfirewall firewall add rule name=DCOM_TCP135 protocol=TCP localport=135 dir=in action=allow
```

3. Add the client application that contains the sink for the callback (`MySQLNotifier.exe`) to the Windows Firewall Exceptions List (use either the Windows Firewall configuration or `NETSH`):

Using the Windows Firewall configuration:

- a. In the Control Panel, double-click **Windows Firewall**.
- b. In the Windows Firewall window's left panel, click **Allow a program or feature through Windows Firewall**.
- c. In the Allowed Programs window, click **Change Settings** and do one of the following:
 - If `MySQLNotifier.exe` is in the Allowed programs and features list, make sure it is checked for the type of networks the computer connects to (Private, Public or both).
 - If `MySQLNotifier.exe` is not in the list, click **Allow another program....**

- i. In the **Add a Program** window, select the `MySQLNotifier.exe` if it exists in the Programs list, otherwise click **Browse...** and go to the directory where `MySQLNotifier.exe` was installed to select it, then click **Add**.
- ii. Make sure `MySQLNotifier.exe` is checked for the type of networks the computer connects to (Private, Public or both).

Using the `NETSH` command:

- a. Open a command prompt window with Administrative rights (you can right-click the Command Prompt icon and click **Run as Administrator**).
- b. Execute the following command, where you change "`[YOUR_INSTALL_DIRECTORY]`":

```
NETSH advfirewall firewall add rule name=MySQLNotifier program=[YOUR_INSTALL_DIRECTORY]\MySQLNotifier.exe
```

4. If Computer B is either a member of `WORKGROUP` or is in a different domain that is untrusted by Computer A, then the callback connection (Connection 2) is created as an Anonymous connection. To grant Anonymous connections DCOM Remote Access permissions:
 - a. Click **Start**, click **Run**, type `DCOMCNFG`, and then click **OK**.
 - b. In the Component Services dialog box, expand Component Services, expand Computers, and then right-click **My Computer** and click **Properties**.
 - c. In the My Computer Properties dialog box, click the **COM Security** tab.
 - d. Under Access Permissions, click **Edit Limits**.
 - e. In the Access Permission dialog box, select **ANONYMOUS LOGON name** in the Group or user names box. In the Allow column under Permissions for User, select **Remote Access**, and then click **OK**.

Monitored Remote Computer (Computer B)

If the user account that is logged on to the computer running the MySQL Notifier (Computer A) is a local administrator on the remote computer (Computer B), such that the same account is an administrator on Computer B, you can skip to the "Allow for remote administration" step.

Setting DCOM security to allow a non-administrator user to access a computer remotely:

1. Grant "DCOM remote launch" and activation permissions for a user or group:
 - a. Click **Start**, click **Run**, type `DCOMCNFG`, and then click **OK**.
 - b. In the Component Services dialog box, expand Component Services, expand Computers, and then right-click **My Computer** and click **Properties**.
 - c. In the My Computer Properties dialog box, click the **COM Security** tab.
 - d. Under Launch and Activation Permission, click **Edit Limits**.
 - e. In the **Launch and Activation Permission** dialog box, follow these steps if your name or your group does not appear in the Groups or user names list:
 - i. In the **Launch and Activation Permission** dialog box, click **Add**.

- ii. In the Select Users or Groups dialog box, add your name and the group in the **Enter the object names to select** box, and then click **OK**.
 - f. In the **Launch and Activation Permission** dialog box, select your user and group in the Group or user names box. In the Allow column under Permissions for User, select **Remote Launch**, select **Remote Activation**, and then click **OK**.
- Grant DCOM remote access permissions:
 - a. Click **Start**, click **Run**, type **DCOMCNFG**, and then click **OK**.
 - b. In the Component Services dialog box, expand Component Services, expand Computers, and then right-click **My Computer** and click **Properties**.
 - c. In the My Computer Properties dialog box, click the **COM Security** tab.
 - d. Under Access Permissions, click **Edit Limits**.
 - e. In the Access Permission dialog box, select **ANONYMOUS LOGON name** in the Group or user names box. In the Allow column under Permissions for User, select **Remote Access**, and then click **OK**.
2. Allowing non-administrator users access to a specific WMI namespace:
 - a. In the Control Panel, double-click **Administrative Tools**.
 - b. In the Administrative Tools window, double-click **Computer Management**.
 - c. In the Computer Management window, expand the **Services and Applications** tree.
 - d. Right-click the WMI Control icon and select **Properties**.
 - e. In the WMI Control Properties window, click the **Security** tab.
 - f. In the Security tab, select the namespace and click **Security**. Root/CIMV2 is a commonly used namespace.
 - g. Locate the appropriate account and check **Remote Enable** in the Permissions list.
3. Allow for remote administration by either editing the **Group Policy Editor** or using **NETSH**:

Using the **Group Policy Editor**:

- a. Click **Start**, click **Run**, type **GPEDIT.MSC**, and then click **OK**.
- b. Under the Local Computer Policy heading, double-click **Computer Configuration**.
- c. Double-click **Administrative Templates**, then **Network**, **Network Connections**, and then **Windows Firewall**.
- d. If the computer is in the domain, then double-click **Domain Profile**; otherwise, double-click **Standard Profile**.
- e. Click **Windows Firewall: Allow inbound remote administration exception**.
- f. On the Action menu either select **Edit**, or double-click the selection from the previous step.

- g. Check the **Enabled** radio button, and then click **OK**.

Using the [NETSH](#) command:

- a. Open a command prompt window with Administrative rights (you can right-click the Command Prompt icon and click Run as Administrator).
- b. Execute the following command:

```
NETSH advfirewall firewall set service RemoteAdmin enable
```

4. Confirm that the user account you are logging in with uses the [Name](#) value and not the [Full Name](#) value:
 - a. In the **Control Panel**, double-click **Administrative Tools**.
 - b. In the **Administrative Tools** window, double-click **Computer Management**.
 - c. In the **Computer Management** window, expand the **System Tools** then **Local Users and Groups**.
 - d. Click the **Users** node, and on the right side panel locate your user and make sure it uses the **Name** value to connect, and not the **Full Name** value.

Common Errors

- [0x80070005](#)
 - DCOM Security was not configured properly (see Computer B, the [Setting DCOM security...](#) step).
 - The remote computer (Computer B) is a member of WORKGROUP or is in a domain that is untrusted by the client computer (Computer A) (see Computer A, the [Grant Anonymous connections DCOM Remote Access permissions](#) step).
- [0x8007000E](#)
 - The remote computer (Computer B) is a member of WORKGROUP or is in a domain that is untrusted by the client computer (Computer A) (see Computer A, the [Grant Anonymous connections DCOM Remote Access permissions](#) step).
- [0x80041003](#)
 - Access to the remote WMI namespace was not configured properly (see Computer B, the [Allowing non-administrator users access to a specific WMI namespace](#) step).
- [0x800706BA](#)
 - The DCOM port is not open on the client computers (Computer A) firewall. See the [Open the DCOM port TCP 135](#) step for Computer A.
 - The remote computer (Computer B) is inaccessible because its network location is set to Public. Make sure you can access it through the Windows Explorer.

1.5 Installing MySQL on Microsoft Windows Using a noinstall Zip Archive

Users who are installing from the [noinstall](#) package can use the instructions in this section to manually install MySQL. The process for installing MySQL from a Zip archive is as follows:

1. Extract the archive to the desired install directory
2. Create an option file
3. Choose a MySQL server type
4. Start the MySQL server
5. Secure the default user accounts

This process is described in the sections that follow.

1.5.1 Extracting the Install Archive

To install MySQL manually, do the following:

1. If you are upgrading from a previous version please refer to [Section 1.8, “Upgrading MySQL on Windows”](#), before beginning the upgrade process.
2. Make sure that you are logged in as a user with administrator privileges.
3. Choose an installation location. Traditionally, the MySQL server is installed in `C:\mysql`. The MySQL Installation Wizard installs MySQL under `C:\Program Files\MySQL`. If you do not install MySQL at `C:\mysql`, you must specify the path to the install directory during startup or in an option file. See [Section 1.5.2, “Creating an Option File”](#).

Note

The MySQL Installer installs MySQL under `C:\Program Files\MySQL`.

4. Extract the install archive to the chosen installation location using your preferred Zip archive tool. Some tools may extract the archive to a folder within your chosen installation location. If this occurs, you can move the contents of the subfolder into the chosen installation location.

1.5.2 Creating an Option File

If you need to specify startup options when you run the server, you can indicate them on the command line or place them in an option file. For options that are used every time the server starts, you may find it most convenient to use an option file to specify your MySQL configuration. This is particularly true under the following circumstances:

- The installation or data directory locations are different from the default locations (`C:\Program Files\MySQL\MySQL Server 5.6` and `C:\Program Files\MySQL\MySQL Server 5.6\data`).
- You need to tune the server settings, such as memory, cache, or InnoDB configuration information.

When the MySQL server starts on Windows, it looks for option files in several locations, such as the Windows directory, `C:\`, and the MySQL installation directory (for the full list of locations, see [Using Option Files](#)). The Windows directory typically is named something like `C:\WINDOWS`. You can determine its exact location from the value of the `WINDIR` environment variable using the following command:

```
C:\> echo %WINDIR%
```

MySQL looks for options in each location first in the `my.ini` file, and then in the `my.cnf` file. However, to avoid confusion, it is best if you use only one file. If your PC uses a boot loader where `C:` is not the boot drive, your only option is to use the `my.ini` file. Whichever option file you use, it must be a plain text file.

Note

When using the MySQL Installer to install MySQL Server, it will create the `my.ini` at the default location. And as of MySQL Server 5.5.27, the user running MySQL Installer is granted full permissions to this new `my.ini`.

In other words, be sure that the MySQL Server user has permission to read the `my.ini` file.

You can also make use of the example option files included with your MySQL distribution; see [Server Configuration Defaults](#).

An option file can be created and modified with any text editor, such as Notepad. For example, if MySQL is installed in `E:\mysql` and the data directory is in `E:\mydata\data`, you can create an option file containing a `[mysqld]` section to specify values for the `basedir` and `datadir` options:

```
[mysqld]
# set basedir to your installation path
basedir=E:/mysql
# set datadir to the location of your data directory
datadir=E:/mydata/data
```

Microsoft Windows path names are specified in option files using (forward) slashes rather than backslashes. If you do use backslashes, double them:

```
[mysqld]
# set basedir to your installation path
basedir=E:\\mysql
# set datadir to the location of your data directory
datadir=E:\\mydata\\data
```

The rules for use of backslash in option file values are given in [Using Option Files](#).

The data directory is located within the `AppData` directory for the user running MySQL.

If you would like to use a data directory in a different location, you should copy the entire contents of the `data` directory to the new location. For example, if you want to use `E:\mydata` as the data directory instead, you must do two things:

1. Move the entire `data` directory and all of its contents from the default location (for example `C:\Program Files\MySQL\MySQL Server 5.6\data`) to `E:\mydata`.
2. Use a `--datadir` option to specify the new data directory location each time you start the server.

1.5.3 Selecting a MySQL Server Type

The following table shows the available servers for Windows in MySQL 5.6.

Binary	Description
<code>mysqld</code>	Optimized binary with named-pipe support

Binary	Description
<code>mysqld-debug</code>	Like <code>mysqld</code> , but compiled with full debugging and automatic memory allocation checking

All of the preceding binaries are optimized for modern Intel processors, but should work on any Intel i386-class or higher processor.

Each of the servers in a distribution support the same set of storage engines. The `SHOW ENGINES` statement displays which engines a given server supports.

All Windows MySQL 5.6 servers have support for symbolic linking of database directories.

MySQL supports TCP/IP on all Windows platforms. MySQL servers on Windows also support named pipes, if you start the server with the `--enable-named-pipe` option. It is necessary to use this option explicitly because some users have experienced problems with shutting down the MySQL server when named pipes were used. The default is to use TCP/IP regardless of platform because named pipes are slower than TCP/IP in many Windows configurations.

1.5.4 Starting the Server for the First Time

This section gives a general overview of starting the MySQL server. The following sections provide more specific information for starting the MySQL server from the command line or as a Windows service.

The information here applies primarily if you installed MySQL using the `Noinstall` version, or if you wish to configure and test MySQL manually rather than with the GUI tools.

Note

The MySQL server will automatically start after using the MySQL Installer, and the `MySQL Notifier` GUI can be used to start/stop/restart at any time.

The examples in these sections assume that MySQL is installed under the default location of `C:\Program Files\MySQL\MySQL Server 5.6`. Adjust the path names shown in the examples if you have MySQL installed in a different location.

Clients have two options. They can use TCP/IP, or they can use a named pipe if the server supports named-pipe connections.

MySQL for Windows also supports shared-memory connections if the server is started with the `--shared-memory` option. Clients can connect through shared memory by using the `--protocol=MEMORY` option.

For information about which server binary to run, see [Section 1.5.3, “Selecting a MySQL Server Type”](#).

Testing is best done from a command prompt in a console window (or “DOS window”). In this way you can have the server display status messages in the window where they are easy to see. If something is wrong with your configuration, these messages make it easier for you to identify and fix any problems.

To start the server, enter this command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld" --console
```

For a server that includes `InnoDB` support, you should see the messages similar to those following as it starts (the path names and sizes may differ):

```
InnoDB: The first specified datafile c:\ibdata\ibdata1 did not exist:
```

```

InnoDB: a new database to be created!
InnoDB: Setting file c:\ibdata\ibdata1 size to 209715200
InnoDB: Database physically writes the file full: wait...
InnoDB: Log file c:\iblogs\ib_logfile0 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile0 size to 31457280
InnoDB: Log file c:\iblogs\ib_logfile1 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile1 size to 31457280
InnoDB: Log file c:\iblogs\ib_logfile2 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile2 size to 31457280
InnoDB: Doublewrite buffer not found: creating new
InnoDB: Doublewrite buffer created
InnoDB: creating foreign key constraint system tables
InnoDB: foreign key constraint system tables created
011024 10:58:25 InnoDB: Started

```

When the server finishes its startup sequence, you should see something like this, which indicates that the server is ready to service client connections:

```

mysqld: ready for connections
Version: '5.6.33' socket: '' port: 3306

```

The server continues to write to the console any further diagnostic output it produces. You can open a new console window in which to run client programs.

If you omit the `--console` option, the server writes diagnostic output to the error log in the data directory (`C:\Program Files\MySQL\MySQL Server 5.6\data` by default). The error log is the file with the `.err` extension, and may be set using the `--log-error` option.

Note

The accounts that are listed in the MySQL grant tables initially have no passwords. After starting the server, you should set up passwords for them using the instructions in [Securing the Initial MySQL Accounts](#).

1.5.5 Starting MySQL from the Windows Command Line

The MySQL server can be started manually from the command line. This can be done on any version of Windows.

Note

The [MySQL Notifier](#) GUI can also be used to start/stop/restart the MySQL server.

To start the `mysqld` server from the command line, you should start a console window (or “DOS window”) and enter this command:

```

C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld"

```

The path to `mysqld` may vary depending on the install location of MySQL on your system.

You can stop the MySQL server by executing this command:

```

C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqladmin" -u root shutdown

```

Note

If the MySQL `root` user account has a password, you need to invoke `mysqladmin` with the `-p` option and supply the password when prompted.

This command invokes the MySQL administrative utility `mysqladmin` to connect to the server and tell it to shut down. The command connects as the MySQL `root` user, which is the default administrative account in the MySQL grant system.

Note

Users in the MySQL grant system are wholly independent from any login users under Microsoft Windows.

If `mysqld` doesn't start, check the error log to see whether the server wrote any messages there to indicate the cause of the problem. By default, the error log is located in the `C:\Program Files\MySQL\MySQL Server 5.6\data` directory. It is the file with a suffix of `.err`, or may be specified by passing in the `--log-error` option. Alternatively, you can try to start the server with the `--console` option; in this case, the server may display some useful information on the screen that will help solve the problem.

The last option is to start `mysqld` with the `--standalone` and `--debug` options. In this case, `mysqld` writes a log file `C:\mysqld.trace` that should contain the reason why `mysqld` doesn't start. See [The DEBUG Package](#).

Use `mysqld --verbose --help` to display all the options that `mysqld` supports.

1.5.6 Customizing the PATH for MySQL Tools

Warning

You must exercise great care when editing your system `PATH` by hand; accidental deletion or modification of any portion of the existing `PATH` value can leave you with a malfunctioning or even unusable system.

To make it easier to invoke MySQL programs, you can add the path name of the MySQL `bin` directory to your Windows system `PATH` environment variable:

- On the Windows desktop, right-click the **My Computer** icon, and select **Properties**.
- Next select the **Advanced** tab from the **System Properties** menu that appears, and click the **Environment Variables** button.
- Under **System Variables**, select **Path**, and then click the **Edit** button. The **Edit System Variable** dialogue should appear.
- Place your cursor at the end of the text appearing in the space marked **Variable Value**. (Use the **End** key to ensure that your cursor is positioned at the very end of the text in this space.) Then enter the complete path name of your MySQL `bin` directory (for example, `C:\Program Files\MySQL\MySQL Server 5.6\bin`)

Note

There must be a semicolon separating this path from any values present in this field.

Dismiss this dialogue, and each dialogue in turn, by clicking **OK** until all of the dialogues that were opened have been dismissed. The new `PATH` value should now be available to any new command shell you open, allowing you to invoke any MySQL executable program by typing its name at the DOS prompt from any directory on the system, without having to supply the path. This includes the servers, the `mysql` client, and all MySQL command-line utilities such as `mysqladmin` and `mysqldump`.

You should not add the MySQL `bin` directory to your Windows `PATH` if you are running multiple MySQL servers on the same machine.

1.5.7 Starting MySQL as a Windows Service

On Windows, the recommended way to run MySQL is to install it as a Windows service, so that MySQL starts and stops automatically when Windows starts and stops. A MySQL server installed as a service can also be controlled from the command line using [NET](#) commands, or with the graphical [Services](#) utility. Generally, to install MySQL as a Windows service you should be logged in using an account that has administrator rights.

Note

The [MySQL Notifier](#) GUI can also be used to monitor the status of the MySQL service.

The [Services](#) utility (the Windows [Service Control Manager](#)) can be found in the Windows Control Panel (under **Administrative Tools** on Windows 2000, XP, Vista, and Server 2003). To avoid conflicts, it is advisable to close the [Services](#) utility while performing server installation or removal operations from the command line.

Installing the service

Before installing MySQL as a Windows service, you should first stop the current server if it is running by using the following command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqladmin"  
-u root shutdown
```

Note

If the MySQL [root](#) user account has a password, you need to invoke [mysqladmin](#) with the [-p](#) option and supply the password when prompted.

This command invokes the MySQL administrative utility [mysqladmin](#) to connect to the server and tell it to shut down. The command connects as the MySQL [root](#) user, which is the default administrative account in the MySQL grant system.

Note

Users in the MySQL grant system are wholly independent from any login users under Windows.

Install the server as a service using this command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld" --install
```

The service-installation command does not start the server. Instructions for that are given later in this section.

To make it easier to invoke MySQL programs, you can add the path name of the MySQL [bin](#) directory to your Windows system [PATH](#) environment variable:

- On the Windows desktop, right-click the **My Computer** icon, and select **Properties**.
- Next select the **Advanced** tab from the **System Properties** menu that appears, and click the **Environment Variables** button.
- Under **System Variables**, select **Path**, and then click the **Edit** button. The **Edit System Variable** dialogue should appear.

- Place your cursor at the end of the text appearing in the space marked **Variable Value**. (Use the **End** key to ensure that your cursor is positioned at the very end of the text in this space.) Then enter the complete path name of your MySQL **bin** directory (for example, `C:\Program Files\MySQL\MySQL Server 5.6\bin`), and there should be a semicolon separating this path from any values present in this field. Dismiss this dialogue, and each dialogue in turn, by clicking **OK** until all of the dialogues that were opened have been dismissed. You should now be able to invoke any MySQL executable program by typing its name at the DOS prompt from any directory on the system, without having to supply the path. This includes the servers, the `mysql` client, and all MySQL command-line utilities such as `mysqladmin` and `mysqldump`.

You should not add the MySQL **bin** directory to your Windows **PATH** if you are running multiple MySQL servers on the same machine.

Warning

You must exercise great care when editing your system **PATH** by hand; accidental deletion or modification of any portion of the existing **PATH** value can leave you with a malfunctioning or even unusable system.

The following additional arguments can be used when installing the service:

- You can specify a service name immediately following the `--install` option. The default service name is `MySQL`.
- If a service name is given, it can be followed by a single option. By convention, this should be `--defaults-file=file_name` to specify the name of an option file from which the server should read options when it starts.

The use of a single option other than `--defaults-file` is possible but discouraged. `--defaults-file` is more flexible because it enables you to specify multiple startup options for the server by placing them in the named option file.

- You can also specify a `--local-service` option following the service name. This causes the server to run using the `LocalService` Windows account that has limited system privileges. This account is available only for Windows XP or newer. If both `--defaults-file` and `--local-service` are given following the service name, they can be in any order.

For a MySQL server that is installed as a Windows service, the following rules determine the service name and option files that the server uses:

- If the service-installation command specifies no service name or the default service name (`MySQL`) following the `--install` option, the server uses the a service name of `MySQL` and reads options from the `[mysqld]` group in the standard option files.
- If the service-installation command specifies a service name other than `MySQL` following the `--install` option, the server uses that service name. It reads options from the `[mysqld]` group and the group that has the same name as the service in the standard option files. This enables you to use the `[mysqld]` group for options that should be used by all MySQL services, and an option group with the service name for use by the server installed with that service name.
- If the service-installation command specifies a `--defaults-file` option after the service name, the server reads options the same way as described in the previous item, except that it reads options only from the named file and ignores the standard option files.

As a more complex example, consider the following command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld"  
--install MySQL --defaults-file=C:\my-opts.cnf
```

Here, the default service name (MySQL) is given after the `--install` option. If no `--defaults-file` option had been given, this command would have the effect of causing the server to read the `[mysqld]` group from the standard option files. However, because the `--defaults-file` option is present, the server reads options from the `[mysqld]` option group, and only from the named file.

Note

On Windows, if the server is started with the `--defaults-file` and `--install` options, `--install` must be first. Otherwise, `mysqld.exe` will attempt to start the MySQL server.

You can also specify options as Start parameters in the Windows [Services](#) utility before you start the MySQL service.

Starting the service

Once a MySQL server has been installed as a service, Windows starts the service automatically whenever Windows starts. The service also can be started immediately from the [Services](#) utility, or by using a `NET START MySQL` command. The `NET` command is not case sensitive.

When run as a service, `mysqld` has no access to a console window, so no messages can be seen there. If `mysqld` does not start, check the error log to see whether the server wrote any messages there to indicate the cause of the problem. The error log is located in the MySQL data directory (for example, `C:\Program Files\MySQL\MySQL Server 5.6\data`). It is the file with a suffix of `.err`.

When a MySQL server has been installed as a service, and the service is running, Windows stops the service automatically when Windows shuts down. The server also can be stopped manually by using the [Services](#) utility, the `NET STOP MySQL` command, or the `mysqladmin shutdown` command.

You also have the choice of installing the server as a manual service if you do not wish for the service to be started automatically during the boot process. To do this, use the `--install-manual` option rather than the `--install` option:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld" --install-manual
```

Removing the service

To remove a server that is installed as a service, first stop it if it is running by executing `NET STOP MySQL`. Then use the `--remove` option to remove it:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld" --remove
```

If `mysqld` is not running as a service, you can start it from the command line. For instructions, see [Section 1.5.5, "Starting MySQL from the Windows Command Line"](#).

If you encounter difficulties during installation, see [Section 1.6, "Troubleshooting a Microsoft Windows MySQL Server Installation"](#).

For more information about stopping or removing a MySQL Windows service, see [Starting Multiple MySQL Instances as Windows Services](#).

1.5.8 Testing The MySQL Installation

You can test whether the MySQL server is working by executing any of the following commands:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqlshow"
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqlshow" -u root mysql
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqladmin" version status proc
C:\> "C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql" test
```

If `mysqld` is slow to respond to TCP/IP connections from client programs, there is probably a problem with your DNS. In this case, start `mysqld` with the `--skip-name-resolve` option and use only `localhost` and IP addresses in the `Host` column of the MySQL grant tables. (Be sure that an account exists that specifies an IP address or you may not be able to connect.)

You can force a MySQL client to use a named-pipe connection rather than TCP/IP by specifying the `--pipe` or `--protocol=PIPE` option, or by specifying `.` (period) as the host name. Use the `--socket` option to specify the name of the pipe if you do not want to use the default pipe name.

If you have set a password for the `root` account, deleted the anonymous account, or created a new user account, then to connect to the MySQL server you must use the appropriate `-u` and `-p` options with the commands shown previously. See [Connecting to the MySQL Server](#).

For more information about `mysqlshow`, see [mysqlshow — Display Database, Table, and Column Information](#).

1.6 Troubleshooting a Microsoft Windows MySQL Server Installation

When installing and running MySQL for the first time, you may encounter certain errors that prevent the MySQL server from starting. This section helps you diagnose and correct some of these errors.

Your first resource when troubleshooting server issues is the [error log](#). The MySQL server uses the error log to record information relevant to the error that prevents the server from starting. The error log is located in the [data directory](#) specified in your `my.ini` file. The default data directory location is `C:\Program Files\MySQL\MySQL Server 5.6\data`, or `C:\ProgramData\MySQL` on Windows 7 and Windows Server 2008. The `C:\ProgramData` directory is hidden by default. You need to change your folder options to see the directory and contents. For more information on the error log and understanding the content, see [The Error Log](#).

For information regarding possible errors, also consult the console messages displayed when the MySQL service is starting. Use the `NET START MySQL` command from the command line after installing `mysqld` as a service to see any error messages regarding the starting of the MySQL server as a service. See [Section 1.5.7, "Starting MySQL as a Windows Service"](#).

The following examples show other common error messages you might encounter when installing MySQL and starting the server for the first time:

- If the MySQL server cannot find the `mysql` privileges database or other critical files, it displays these messages:

```
System error 1067 has occurred.
Fatal error: Can't open and lock privilege tables:
Table 'mysql.user' doesn't exist
```

These messages often occur when the MySQL base or data directories are installed in different locations than the default locations (`C:\Program Files\MySQL\MySQL Server 5.6` and `C:\Program Files\MySQL\MySQL Server 5.6\data`, respectively).

This situation can occur when MySQL is upgraded and installed to a new location, but the configuration file is not updated to reflect the new location. In addition, old and new configuration files might conflict. Be sure to delete or rename any old configuration files when upgrading MySQL.

If you have installed MySQL to a directory other than `C:\Program Files\MySQL\MySQL Server 5.6`, ensure that the MySQL server is aware of this through the use of a configuration (`my.ini`) file. Put the `my.ini` file in your Windows directory, typically `C:\WINDOWS`. To determine its exact location from the value of the `WINDIR` environment variable, issue the following command from the command prompt:

```
C:\> echo %WINDIR%
```

You can create or modify an option file with any text editor, such as Notepad. For example, if MySQL is installed in `E:\mysql` and the data directory is `D:\MySQLdata`, you can create the option file and set up a `[mysqld]` section to specify values for the `basedir` and `datadir` options:

```
[mysqld]
# set basedir to your installation path
basedir=E:/mysql
# set datadir to the location of your data directory
datadir=D:/MySQLdata
```

Microsoft Windows path names are specified in option files using (forward) slashes rather than backslashes. If you do use backslashes, double them:

```
[mysqld]
# set basedir to your installation path
basedir=C:\\Program Files\\MySQL\\MySQL Server 5.6
# set datadir to the location of your data directory
datadir=D:\\MySQLdata
```

The rules for use of backslash in option file values are given in [Using Option Files](#).

If you change the `datadir` value in your MySQL configuration file, you must move the contents of the existing MySQL data directory before restarting the MySQL server.

See [Section 1.5.2, "Creating an Option File"](#).

- If you reinstall or upgrade MySQL without first stopping and removing the existing MySQL service and install MySQL using the MySQL Installer, you might see this error:

```
Error: Cannot create Windows service for MySql. Error: 0
```

This occurs when the Configuration Wizard tries to install the service and finds an existing service with the same name.

One solution to this problem is to choose a service name other than `mysql` when using the configuration wizard. This enables the new service to be installed correctly, but leaves the outdated service in place. Although this is harmless, it is best to remove old services that are no longer in use.

To permanently remove the old `mysql` service, execute the following command as a user with administrative privileges, on the command line:

```
C:\> sc delete mysql
[SC] DeleteService SUCCESS
```

If the `sc` utility is not available for your version of Windows, download the `deLSrv` utility from <http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/delsrv-o.asp> and use the `deLSrv mysql` syntax.

1.7 Windows Postinstallation Procedures

GUI tools exist that perform most of the tasks described in this section, including:

- [MySQL Installer](#): Used to install and upgrade MySQL products.
- [MySQL Workbench](#): Manages the MySQL server and edits SQL statements.
- [MySQL Notifier](#): Starts, stops, or restarts the MySQL server, and monitors its status.
- [MySQL for Excel](#): Edits MySQL data with Microsoft Excel.

On Windows, you need not create the data directory and the grant tables. MySQL Windows distributions include the grant tables with a set of preinitialized accounts in the `mysql` database under the data directory.

Regarding passwords, if you installed MySQL using the MySQL Installer, you may have already assigned passwords to the accounts. (See [Section 1.3, “Installing MySQL on Microsoft Windows Using MySQL Installer”](#).) Otherwise, use the password-assignment procedure given in [Securing the Initial MySQL Accounts](#).

Before assigning passwords, you might want to try running some client programs to make sure that you can connect to the server and that it is operating properly. Make sure that the server is running (see [Section 1.5.4, “Starting the Server for the First Time”](#)). You can also set up a MySQL service that runs automatically when Windows starts (see [Section 1.5.7, “Starting MySQL as a Windows Service”](#)).

These instructions assume that your current location is the MySQL installation directory and that it has a `bin` subdirectory containing the MySQL programs used here. If that is not true, adjust the command path names accordingly.

If you installed MySQL using MySQL Installer (see [Section 1.3, “Installing MySQL on Microsoft Windows Using MySQL Installer”](#)), the default installation directory is `C:\Program Files\MySQL\MySQL Server 5.6`:

```
C:\> cd "C:\Program Files\MySQL\MySQL Server 5.6"
```

A common installation location for installation from a Zip package is `C:\mysql`:

```
C:\> cd C:\mysql
```

Alternatively, add the `bin` directory to your `PATH` environment variable setting. That enables your command interpreter to find MySQL programs properly, so that you can run a program by typing only its name, not its path name. See [Section 1.5.6, “Customizing the PATH for MySQL Tools”](#).

With the server running, issue the following commands to verify that you can retrieve information from the server. The output should be similar to that shown here.

Use `mysqlshow` to see what databases exist:

```
C:\> bin\mysqlshow
+-----+
| Databases |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| test           |
```

The list of installed databases may vary, but will always include the minimum of `mysql` and `information_schema`.

The preceding command (and commands for other MySQL programs such as `mysql`) may not work if the correct MySQL account does not exist. For example, the program may fail with an error, or you may not be able to view all databases. If you installed MySQL using MySQL Installer, the `root` user will have been created automatically with the password you supplied. In this case, you should use the `-u root` and `-p` options. (You must use those options if you have already secured the initial MySQL accounts.) With `-p`, the client program prompts for the `root` password. For example:

```
C:\> bin\mysqlshow -u root -p
Enter password: (enter root password here)
+-----+
|      Databases      |
+-----+
| information_schema |
| mysql              |
| performance_schema |
| test               |
+-----+
```

If you specify a database name, `mysqlshow` displays a list of the tables within the database:

```
C:\> bin\mysqlshow mysql
Database: mysql
+-----+
|      Tables      |
+-----+
| columns_priv      |
| db                 |
| event             |
| func              |
| general_log        |
| help_category      |
| help_keyword       |
| help_relation      |
| help_topic         |
| innodb_index_stats |
| innodb_table_stats |
| ndb_binlog_index   |
| plugin             |
| proc               |
| procs_priv         |
| proxies_priv       |
| servers            |
| slave_master_info   |
| slave_relay_log_info |
| slave_worker_info  |
| slow_log           |
| tables_priv        |
| time_zone          |
| time_zone_leap_second |
| time_zone_name     |
| time_zone_transition |
| time_zone_transition_type |
| user               |
+-----+
```

Use the `mysql` program to select information from a table in the `mysql` database:


```
C:\> bin\mysql -e "SELECT User, Host, plugin FROM mysql.user" mysql
+-----+-----+-----+
| User | Host | plugin |
+-----+-----+-----+
| root | localhost | mysql_native_password |
+-----+-----+-----+
```

For more information about `mysql` and `mysqlshow`, see [mysql — The MySQL Command-Line Tool](#), and [mysqlshow — Display Database, Table, and Column Information](#).

1.8 Upgrading MySQL on Windows

To upgrade MySQL on Windows, follow these steps:

1. Review [Upgrading MySQL](#), for additional information on upgrading MySQL that is not specific to Windows.
2. Always back up your current MySQL installation before performing an upgrade. See [Database Backup Methods](#).
3. Download the latest Windows distribution of MySQL from <http://dev.mysql.com/downloads/>.
4. Before upgrading MySQL, stop the server. If the server is installed as a service, stop the service with the following command from the command prompt:

```
C:\> NET STOP MySQL
```

If you are not running the MySQL server as a service, use `mysqladmin` to stop it. For example, before upgrading from MySQL 5.5 to 5.6, use `mysqladmin` from MySQL 5.5 as follows:

```
C:\> "C:\Program Files\MySQL\MySQL Server 5.5\bin\mysqladmin" -u root shutdown
```

Note

If the MySQL `root` user account has a password, invoke `mysqladmin` with the `-p` option and enter the password when prompted.

5. Before upgrading to MySQL 5.6 from a version previous to 4.1.5, or from a version of MySQL installed from a Zip archive to a version of MySQL installed with the MySQL Installation Wizard, you must first manually remove the previous installation and MySQL service (if the server is installed as a service).

To remove the MySQL service, use the following command:

```
C:\> C:\mysql\bin\mysqld --remove
```

If you do not remove the existing service, the MySQL Installation Wizard may fail to properly install the new MySQL service.

6. If you are using the MySQL Installer, start it as described in [Section 1.3, “Installing MySQL on Microsoft Windows Using MySQL Installer”](#).
7. If you are upgrading MySQL from a Zip archive, extract the archive. You may either overwrite your existing MySQL installation (usually located at `C:\mysql`), or install it into a different directory, such as `C:\mysql5`. Overwriting the existing installation is recommended. However, for upgrades (as opposed to installing for the first time), you must remove the data directory from your existing MySQL installation to avoid replacing your current data files. To do so, follow these steps:

- a. Unzip the Zip archive in some location other than your current MySQL installation
- b. Remove the data directory
- c. Rezip the Zip archive
- d. Unzip the modified Zip archive on top of your existing installation

Alternatively:

- a. Unzip the Zip archive in some location other than your current MySQL installation
 - b. Remove the data directory
 - c. Move the data directory from the current MySQL installation to the location of the just-removed data directory
 - d. Remove the current MySQL installation
 - e. Move the unzipped installation to the location of the just-removed installation
8. If you were running MySQL as a Windows service and you had to remove the service earlier in this procedure, reinstall the service. (See [Section 1.5.7, “Starting MySQL as a Windows Service”](#).)
 9. Restart the server. For example, use `NET START MySQL` if you run MySQL as a service, or invoke `mysqld` directly otherwise.
 10. As Administrator, run `mysql_upgrade` to check your tables, attempt to repair them if necessary, and update your grant tables if they have changed so that you can take advantage of any new capabilities. See [mysql_upgrade — Check and Upgrade MySQL Tables](#).
 11. If you encounter errors, see [Section 1.6, “Troubleshooting a Microsoft Windows MySQL Server Installation”](#).

Chapter 2 Connection to MySQL Server Failing on Windows

When you're running a MySQL server on Windows with many TCP/IP connections to it, and you're experiencing that quite often your clients get a [Can't connect to MySQL server](#) error, the reason might be that Windows does not allow for enough ephemeral (short-lived) ports to serve those connections.

The purpose of [TIME_WAIT](#) is to keep a connection accepting packets even after the connection has been closed. This is because Internet routing can cause a packet to take a slow route to its destination and it may arrive after both sides have agreed to close. If the port is in use for a new connection, that packet from the old connection could break the protocol or compromise personal information from the original connection. The [TIME_WAIT](#) delay prevents this by ensuring that the port cannot be reused until after some time has been permitted for those delayed packets to arrive.

It is safe to reduce [TIME_WAIT](#) greatly on LAN connections because there is little chance of packets arriving at very long delays, as they could through the Internet with its comparatively large distances and latencies.

Windows permits ephemeral (short-lived) TCP ports to the user. After any port is closed it will remain in a [TIME_WAIT](#) status for 120 seconds. The port will not be available again until this time expires. The default range of port numbers depends on the version of Windows, with a more limited number of ports in older versions:

- Windows through Server 2003: Ports in range 1025–5000
- Windows Vista, Server 2008, and newer: Ports in range 49152–65535

With a small stack of available TCP ports (5000) and a high number of TCP ports being open and closed over a short period of time along with the [TIME_WAIT](#) status you have a good chance for running out of ports. There are two ways to address this problem:

- Reduce the number of TCP ports consumed quickly by investigating connection pooling or persistent connections where possible
- Tune some settings in the Windows registry (see below)

Important

The following procedure involves modifying the Windows registry. Before you modify the registry, make sure to back it up and make sure that you understand how to restore it if a problem occurs. For information about how to back up, restore, and edit the registry, view the following article in the Microsoft Knowledge Base: <http://support.microsoft.com/kb/256986/EN-US/>.

1. Start Registry Editor ([Regedt32.exe](#)).
2. Locate the following key in the registry:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

3. On the [Edit](#) menu, click [Add Value](#), and then add the following registry value:

```
Value Name: MaxUserPort
Data Type: REG_DWORD
Value: 65534
```

This sets the number of ephemeral ports available to any user. The valid range is between 5000 and 65534 (decimal). The default value is 0x1388 (5000 decimal).

-
4. On the **Edit** menu, click **Add Value**, and then add the following registry value:

```
Value Name: TcpTimedWaitDelay  
Data Type: REG_DWORD  
Value: 30
```

This sets the number of seconds to hold a TCP port connection in **TIME_WAIT** state before closing. The valid range is between 30 and 300 decimal, although you may wish to check with Microsoft for the latest permitted values. The default value is 0x78 (120 decimal).

5. Quit Registry Editor.
6. Reboot the machine.

Note

Undoing the above should be as simple as deleting the registry entries you've created.

Chapter 3 Resetting the Root Password: Windows Systems

On Windows, use the following procedure to reset the password for the MySQL `'root'@'localhost'` account. To change the password for a `root` account with a different host name part, modify the instructions to use that host name.

1. Log on to your system as Administrator.
2. Stop the MySQL server if it is running. For a server that is running as a Windows service, go to the Services manager: From the **Start** menu, select **Control Panel**, then **Administrative Tools**, then **Services**. Find the MySQL service in the list and stop it.

If your server is not running as a service, you may need to use the Task Manager to force it to stop.

3. Create a text file containing the following statement on a single line. Replace the password with the password that you want to use.

```
SET PASSWORD FOR 'root'@'localhost' = PASSWORD('MyNewPass');
```

4. Save the file. This example assumes that you name the file `C:\mysql-init.txt`.
5. Open a console window to get to the command prompt: From the **Start** menu, select **Run**, then enter `cmd` as the command to be run.
6. Start the MySQL server with the special `--init-file` option (notice that the backslash in the option value is doubled):

```
C:\> cd "C:\Program Files\MySQL\MySQL Server 5.6\bin"
C:\> mysqld --init-file=C:\mysql-init.txt
```

If you installed MySQL to a different location, adjust the `cd` command accordingly.

The server executes the contents of the file named by the `--init-file` option at startup, changing the `'root'@'localhost'` account password.

To have server output to appear in the console window rather than in a log file, add the `--console` option to the `mysqld` command.

If you installed MySQL using the MySQL Installation Wizard, you may need to specify a `--defaults-file` option. For example:

```
C:\> mysqld
      --defaults-file="C:\\ProgramData\\MySQL\\MySQL Server 5.6\\my.ini"
      --init-file=C:\\mysql-init.txt
```

The appropriate `--defaults-file` setting can be found using the Services Manager: From the **Start** menu, select **Control Panel**, then **Administrative Tools**, then **Services**. Find the MySQL service in the list, right-click it, and choose the **Properties** option. The **Path to executable** field contains the `--defaults-file` setting.

7. After the server has started successfully, delete `C:\mysql-init.txt`.

You should now be able to connect to the MySQL server as `root` using the new password. Stop the MySQL server and restart it normally. If you run the server as a service, start it from the Windows Services window. If you start the server manually, use whatever command you normally use.

Chapter 4 MySQL for Excel

Chapter 5 Installation

MySQL for Excel is a product for Windows, and it is installed with MySQL Installer. And typically you will not be required to install or configure additional tools to use MySQL for Excel.

MySQL for Excel Requirements

The MySQL Installer installation process will check if these requirements are met, or notify you if further action is required before proceeding with the installation.

- *.NET Framework 4.0* (Client or Full Profile).
- *Microsoft Office Excel 2007* or greater, for Microsoft Windows.
- *Visual Studio 2010 Tools for Office Runtime*, and MySQL Installer may install this for you.

Note

This requirement is different than *Office Developer Tools for Visual Studio*, which is not a substitute.

- An available MySQL Server connection.

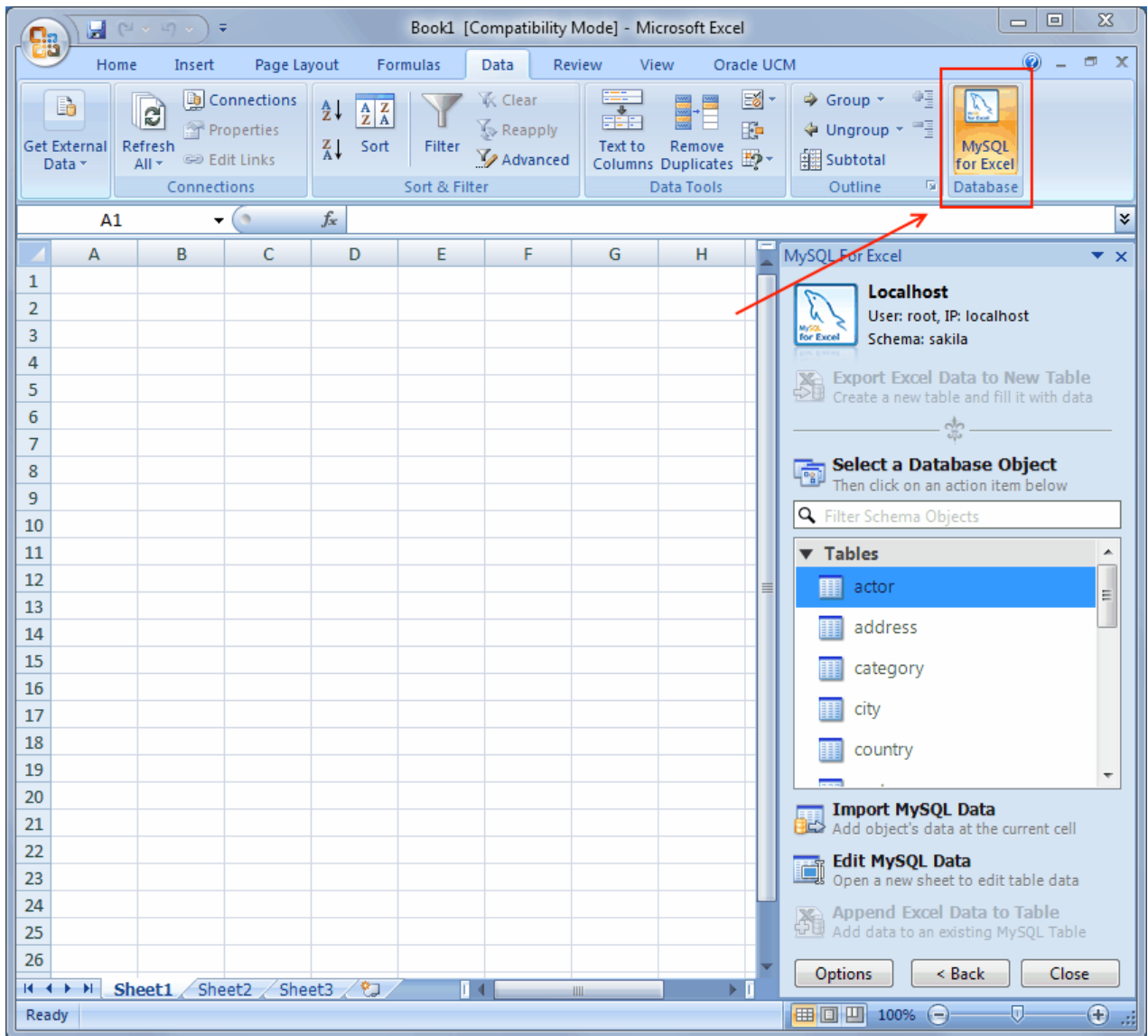
MySQL for Excel Download

Either install MySQL for Excel using the MySQL Installer for Windows (a system that manages installations and updates for all MySQL products on Windows), or download and execute the standalone file. The download links are as follows:

- **MySQL Installer:** Download and execute the [MySQL Installer MSI](#) file. Select the MySQL for Excel product and then proceed with the installation. See the [MySQL Installer manual](#) for additional details. This is the recommended approach.
- **Standalone:** Download and execute the [MySQL for Excel standalone MSI](#) file.

MySQL for Excel is loaded and executed by selecting the **Data** menu tab in Excel, and then choosing the "MySQL for Excel" Database icon. This opens a new Excel sidebar with the available MySQL for Excel options. The navigation bar with the MySQL for Excel icon is shown in the following screenshot:

Figure 5.1 The MySQL for Excel navigation bar



Chapter 6 Configuration

Table of Contents

6.1 Global Options and Preferences 69

6.2 Managing MySQL Connections 73

This section is divided into two overlapping topics; configuring the [global options](#), and managing the [MySQL connections](#).

6.1 Global Options and Preferences

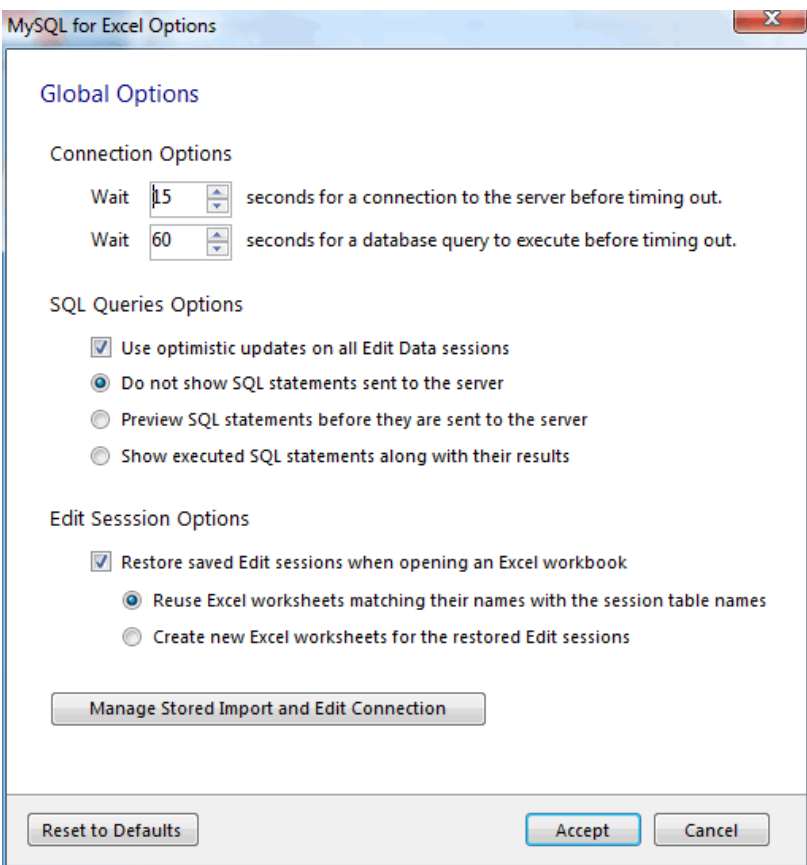
Each action, such as **Import MySQL Data**, has its own set of options. The buttons on these pages include:

Actions include:

- Clicking **Accept**: Saves option changes to your host, and preserves these changes across all sessions and future Excel instances.
- Clicking **Reset to Defaults**: Resets all option values on the current options window to their default settings. Click **Accept** to save the changes.

A set of "global" options affect the entire plugin, as described here:

Figure 6.1 The MySQL for Excel configuration: Global Options

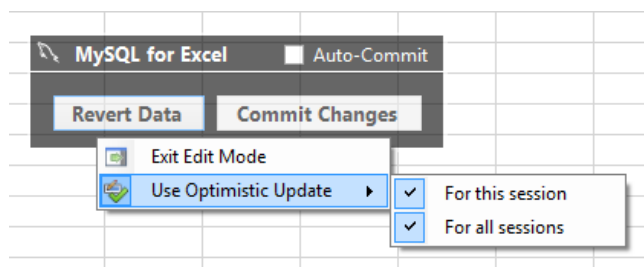


- Connection Options:

- Wait [] seconds for a connection to the server before timing out. Defaults to 15.
- Wait [] seconds for a database query to execute before timing out. Defaults to 60.
- SQL Queries Options:
 - [] Use optimistic updates on all Edit Data sessions. This option helps prevent unintentional data overwrite, in that it checks for external edits before committing your changes. For example, between the time you loaded the data into Excel, made changes in Excel, and committed, a different user could have edited the same cells elsewhere in MySQL using MySQL Workbench or some other means. The optimistic updates feature checks for these changes, and notifies you accordingly.

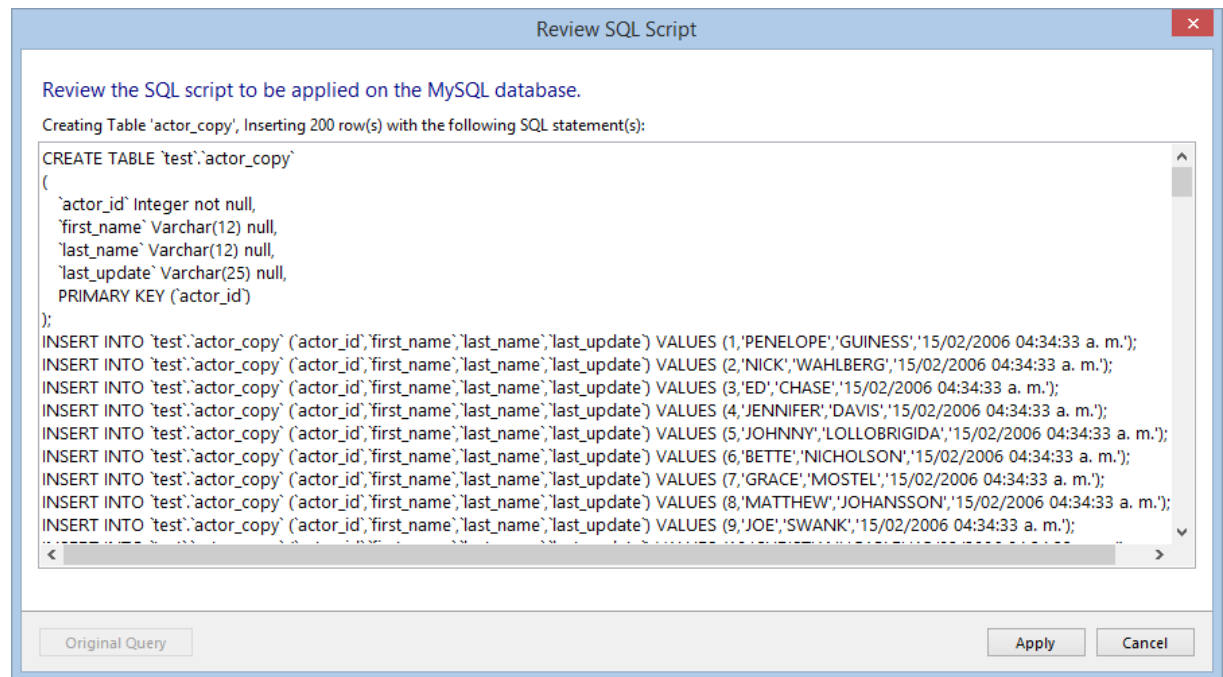
Optimistic updates can also be configured at runtime for all Edit Sessions, or for a specific edit session by right-clicking the **Edit Session** floating dialog and choosing the desired **Use Optimistic Update** option, as demonstrated below:

Figure 6.2 Optimistic Updates: Configuring at Runtime



This option is enabled by default.

- () Do not show SQL statements sent to the server: When enabled, SQL statements are now displayed, and only their results are displayed in the information dialog. Enabled by default.
- () Preview SQL statements before they are sent to the server: When enabled, it adds an extra step to the Create Schema, Export Data, Append Data and Edit Data operations before a statement is committed to the server. It enables the "Review SQL Script" dialog, as shown below for an "Export Data" operation:

Figure 6.3 The MySQL for Excel configuration: Preview Option

From here you can modify the SQL statements before they are executed, which also enables the **Original Query** button. If clicked, it will revert all modifications to the script to restore the SQL to its original form (when the dialog was first opened).

This option is disabled by default.

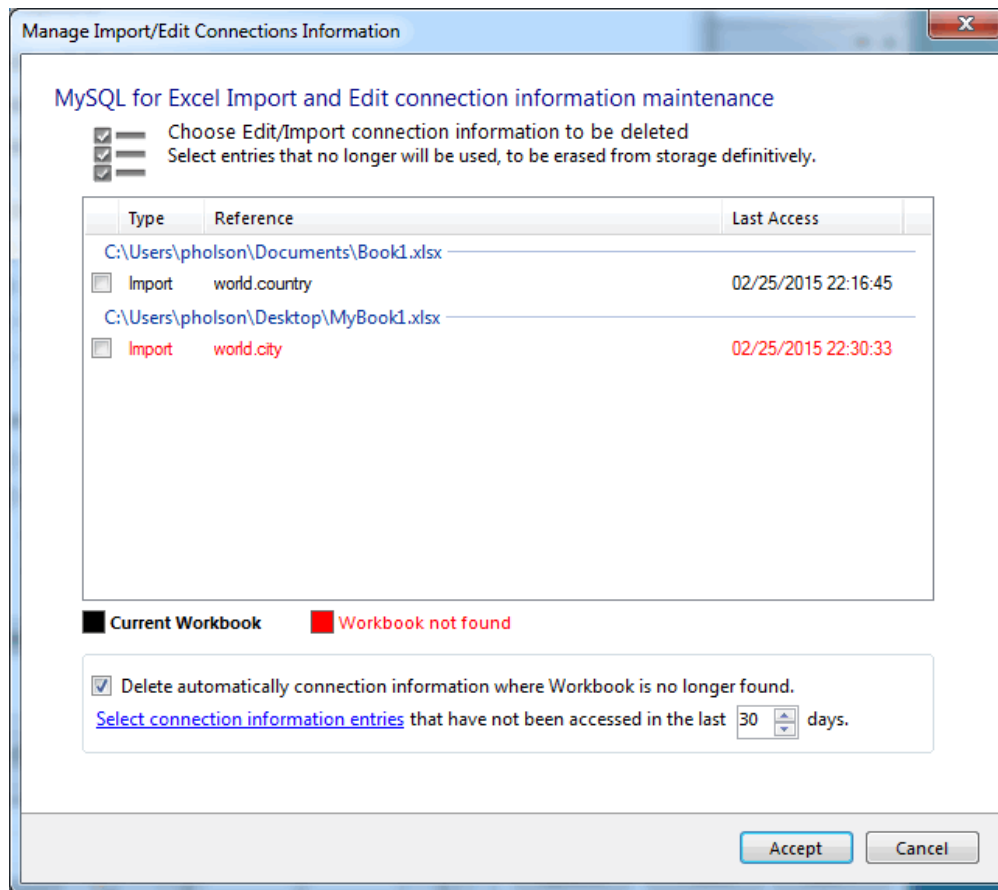
- ☐ *Show executed SQL statements along with their results:* When enabled, SQL statements are first executed and then the information dialog includes both the results and the executed statements. This is helpful when reviewing the recently executed queries when comparing the results.

This option is disabled by default.

- **Edit Session Options:**

- ☐ Restore saved Edit sessions when opening an Excel workbook. Enabled by default.
- ☐ Reuse Excel worksheets matching their names with the session table names. Enabled by default.
- ☐ Create new Excel worksheets for the restored Edit sessions. Disabled by default.

- **Manage Stored Import and Edit Connections:** See a list of saved Excel files with linked MySQL connections.

Figure 6.4 MySQL for Excel: Manage Stored MySQL Connections

This lists the connected Excel worksheets that are known to MySQL for Excel. From here you can view these connections, and optionally delete them. By default, clicking **Apply** will delete connections to missing worksheets but this behavior is configurable. Additionally, clicking the *Select connection information entries* link checks (selects for deletion) books that you have not accessed for n days, where n defaults to 30.

Note

This option was added in MySQL for Excel 1.3.0

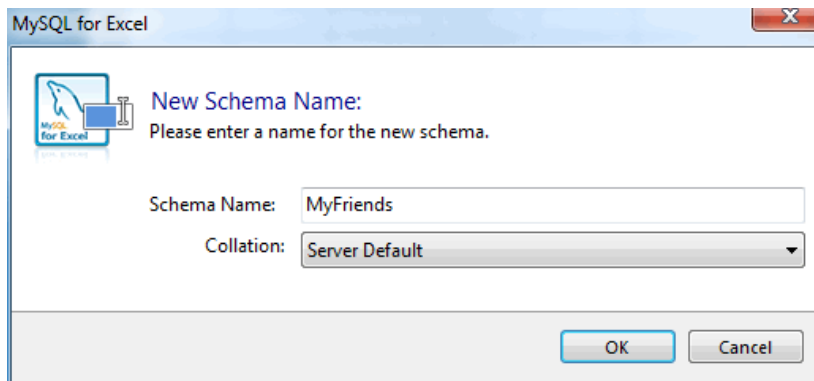
Note

The options to automatically delete missing connections, or delete connections not accessed for n days, were added in MySQL for Excel 1.3.4.

For additional information about managing MySQL connections using MySQL for Excel, see [Section 6.2, "Managing MySQL Connections"](#).

Adding a New Schema

Select a MySQL connection and then click **Create New Schema** from the MySQL for Excel toolbar to add a new and empty MySQL schema.

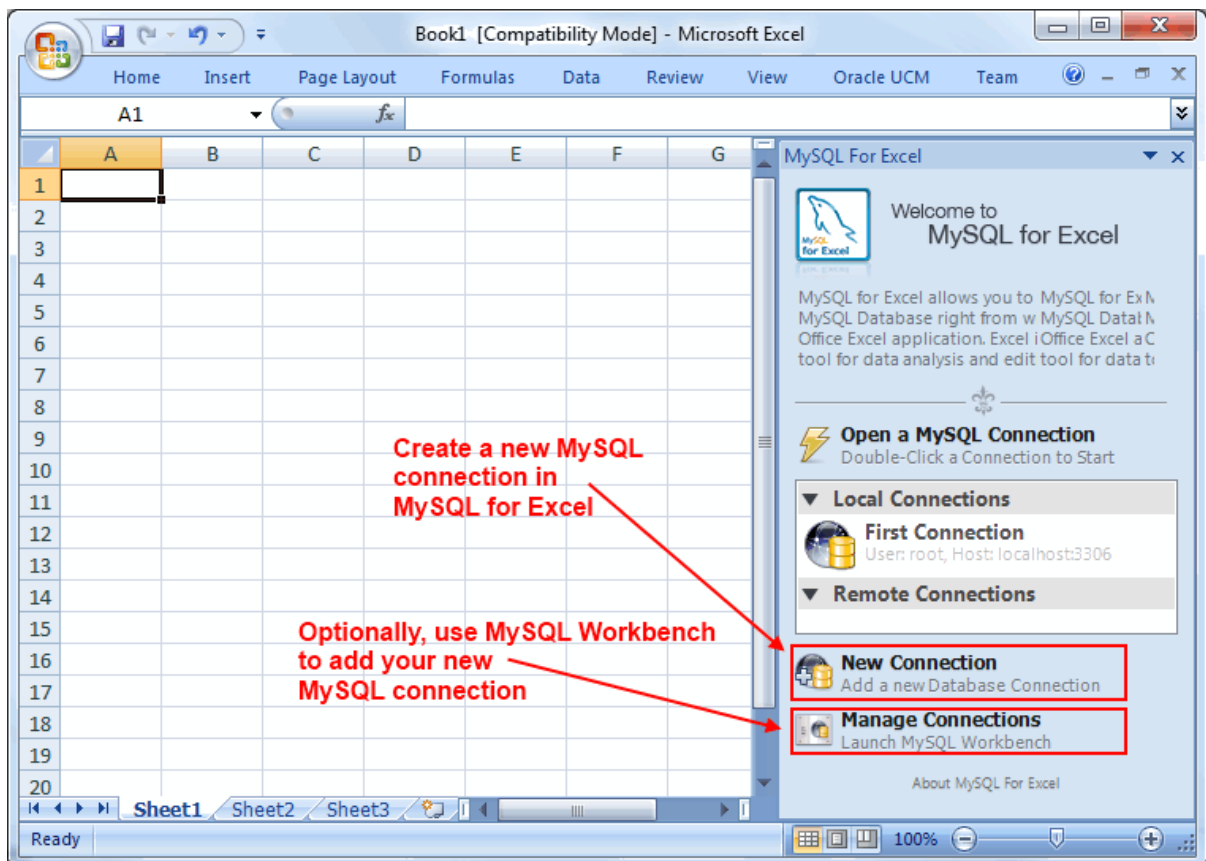
Figure 6.5 MySQL for Excel: Adding a New Schema

6.2 Managing MySQL Connections

MySQL for Excel shares its MySQL connections with MySQL Workbench, although it is optional to have MySQL Workbench installed. Creating and editing MySQL connections in either application will edit the MySQL connection information for both applications.

Adding MySQL Connections

You can use MySQL for Excel or MySQL Workbench to add new MySQL connections.

Figure 6.6 MySQL for Excel: Add a New MySQL Connection

From Excel, click **New Connection** to open the new connection dialog as demonstrated in the following partially filled screenshot:

Figure 6.7 MySQL for Excel: Add a New MySQL Connection Dialog

The screenshot shows the 'MySQL Instance Connection' dialog box. It is divided into several sections:

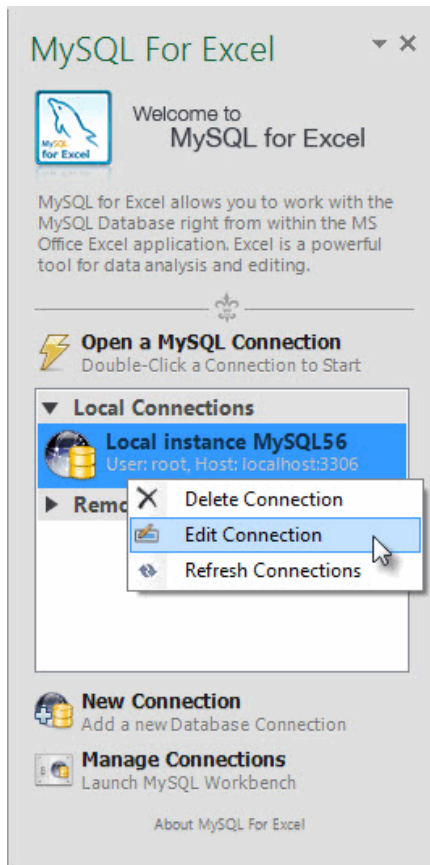
- Connection Name:** A text box containing 'My New Connection Name' with a hint 'Type a name for the connection'.
- Connection Method:** A dropdown menu set to 'Standard (TCP/IP)' with a hint 'Method to use to connect to the RDBMS'.
- Connection Status:** A green question mark icon followed by the text 'Unknown'.
- Parameters:** A group box containing:
 - Hostname:** 'localhost' with a hint 'Name or IP address of the server host - TCP/IP port.'
 - Port:** '3306' with a hint 'Name or IP address of the server host - TCP/IP port.'
 - Username:** 'example' with a hint 'Name of the user to connect with.'
 - Password:** Masked with dots, with a hint 'The user's password, stored in a secured vault.'
 - Default Schema:** 'sakila' with a hint 'The default schema, leave blank to select it later.'
- Advanced:** A group box containing two checkboxes:
 - ☐ Use ANSI quotes to quote identifiers. (Hint: 'If enabled this option overwrites the server side settings.')
 - ☐ Use compression protocol. (Hint: 'Select this option for WAN connections')
- Buttons:** 'Test Connection', 'Cancel', and 'OK' at the bottom.

Fill out the connection details, click **Test Connection** to confirm the MySQL connection is valid, and click **OK** to save the new connection.

Editing MySQL Connections

Editing MySQL Connections in MySQL for Excel

To edit a MySQL connection, right-click the connection you want to modify and select **Edit Connection** from the context menu, like so:

Figure 6.8 MySQL for Excel: Choosing a MySQL Connection to Edit

The MySQL connection edit dialog is similar to the edit dialog in MySQL Workbench. Configure the changes and click **OK** to save your changes:

Figure 6.9 MySQL for Excel: Editing a MySQL Connection

MySQL Instance Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Connection Status: ? Unknown

Parameters

Hostname: Port: Name or IP address of the server host - TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password, stored in a secured vault.

Default Schema: The default schema, leave blank to select it later.

Advanced

☐ Use ANSI quotes to quote identifiers. If enabled this option overwrites the server side settings.

☐ Use compression protocol. Select this option for WAN connections

Editing MySQL Connections in MySQL Workbench

Optionally, you can edit your MySQL for Excel MySQL connections using MySQL Workbench. To do this, open MySQL Workbench, edit a MySQL connection, and then refresh the connection list in MySQL for Excel.

For information about editing MySQL connections in MySQL Workbench, see the MySQL Workbench documentation titled [MySQL Connections](#).

Delete MySQL Connections

MySQL connections can be deleted from MySQL for Excel or MySQL Workbench.

Note

MySQL connections cannot be deleted if MySQL Workbench is open. To remove connections, you must first close MySQL Workbench.

To delete an edit or import connection from MySQL for Excel to a particular Excel worksheet, click **Options, Manage Stored Import and Edit Connection**, check the desired worksheets, and then click **Apply** to execute the delete action.

Chapter 7 What Is New In MySQL for Excel

Table of Contents

7.1 What Is New In MySQL for Excel 1.3	77
7.2 What Is New In MySQL for Excel 1.2	77

This section summarizes how MySQL for Excel progressed with each minor and major release.

7.1 What Is New In MySQL for Excel 1.3

Most of the new features added to MySQL for Excel 1.3.x involve improvements to the **Data Import** functionality.

- You can now refresh imported data from the source MySQL database by clicking **Refresh** from the context-menu, or **Refresh All** from the navigation menu. These check for changes in the source MySQL database and update your imported MySQL data accordingly.

Use case: A colleague sends you a MySQL Excel spreadsheet with data exported from a MySQL database. You open the file several days later, and worry that the data is outdated so you click **Refresh**.

- A new **Refresh To Defaults** button was added to the options pages. It changes each option to the default value, and you then confirm (or cancel) the application of these changes.
- Enabling the new **Add Summary Fields for Numeric Columns** option adds a summary field to the end of each numeric column in Excel. From here, you choose the desired function for the column, such as total or average.
- You may now import data from multiple objects in a single operation. Use **Control** or **Shift** to select multiple objects (tables and/or views) from the MySQL for Excel panel, and click **Import** to open the new dialog for selecting additional objects that have direct relationships to the objects you selected. Each object opens in its own Worksheet.

From this new dialog, you may also generate a **Relationships** model in Excel. This functionality requires Excel 2013 or higher, or Excel 2010 with the PowerPivot add-in.

- A new **Create a PivotTable with the Imported Data** option was added. This creates a Pivot Table in Excel.
- All options now have descriptive tooltips. Hover over an option/preference to view helpful information about its use.
- You may now specify a collation for created schemas. The collation type defaults to "Server Default." These statements can be reviewed before execution.
- All MySQL data types are now available when performing **Data Export** operations. By default, only the most commonly used data types are listed, which was only behavior in previous versions of MySQL for Excel. You may still type in a type instead of selecting it.

7.2 What Is New In MySQL for Excel 1.2

- **Edit Connections:** MySQL connections can now be edited from within the MySQL for Excel plugin by right-clicking and choosing **Edit Connection**. Before, these connections could only be edited with MySQL Workbench.

- **Optimistic Updates:** Previously, only "Pessimistic Updates" were used, which means that pressing **Commit Changes** would overwrite changes performed outside of MySQL for Excel for the edited cells.

Both options remain available today, and optimistic updates are enabled by default. This update type can be set either as a preference, or toggled per session.

- The **Append Data** dialog will now notify you of incompatible types (with visual warnings) when mapping source Excel columns to target MySQL columns.

If a mismatch is discovered, then the column in the source grid that contains the mapped Excel data turns red, and selecting this column displays a warning with text explaining that the source data is not suitable for the mapped target column's data type.

- New preview preferences allow you to enable one of the following three options:
 - **Preview SQL statements before they are sent to the Server:** View (and optionally) edit the MySQL UPDATE/INSERT statements before they are committed.
 - **Show executed SQL statements along with the results:** View the statements after they are committed, which is the current behavior.
 - **Do not show the MySQL statements:** Only show summary information, such as number of affected rows, and not MySQL statements. This is enabled by default.
- **Create Table:** The **Data Export** feature now has the option to only create the table without inserting the data.

To execute, toggle the **Export Data** button to **Create Table**, and then click.

- The selected schema name is now displayed on top of the MySQL for Excel Database Object Selection panel.
- The **Advanced Options** dialogs opened from the Import, Export and Append Data windows now immediately apply the option changes, when before the **Advanced Options** dialog had to be reopened before the changes could be previewed.
- **Edit Data** sessions can now be saved: Using the new **Edit Session** preferences, these sessions were automatically closed after closing an Excel workbook. This data, such as the Workbench connection ID, MySQL schema, and MySQL table name, can now be preserved if the Excel workbook is saved to disk, and available when the Excel workbook is reopened.
- Excel tables are automatically created for any data imported from MySQL to an Excel worksheet, with a name like "Schema.DB-Object-name". The DB object name can be a MySQL table, view, or stored procedure. Options for this feature are listed under **Import Data, Advanced Options**. The newly created Excel tables can be referenced for data analysis in Pivot Tables or reports.

Chapter 8 Edit MySQL Data in Excel

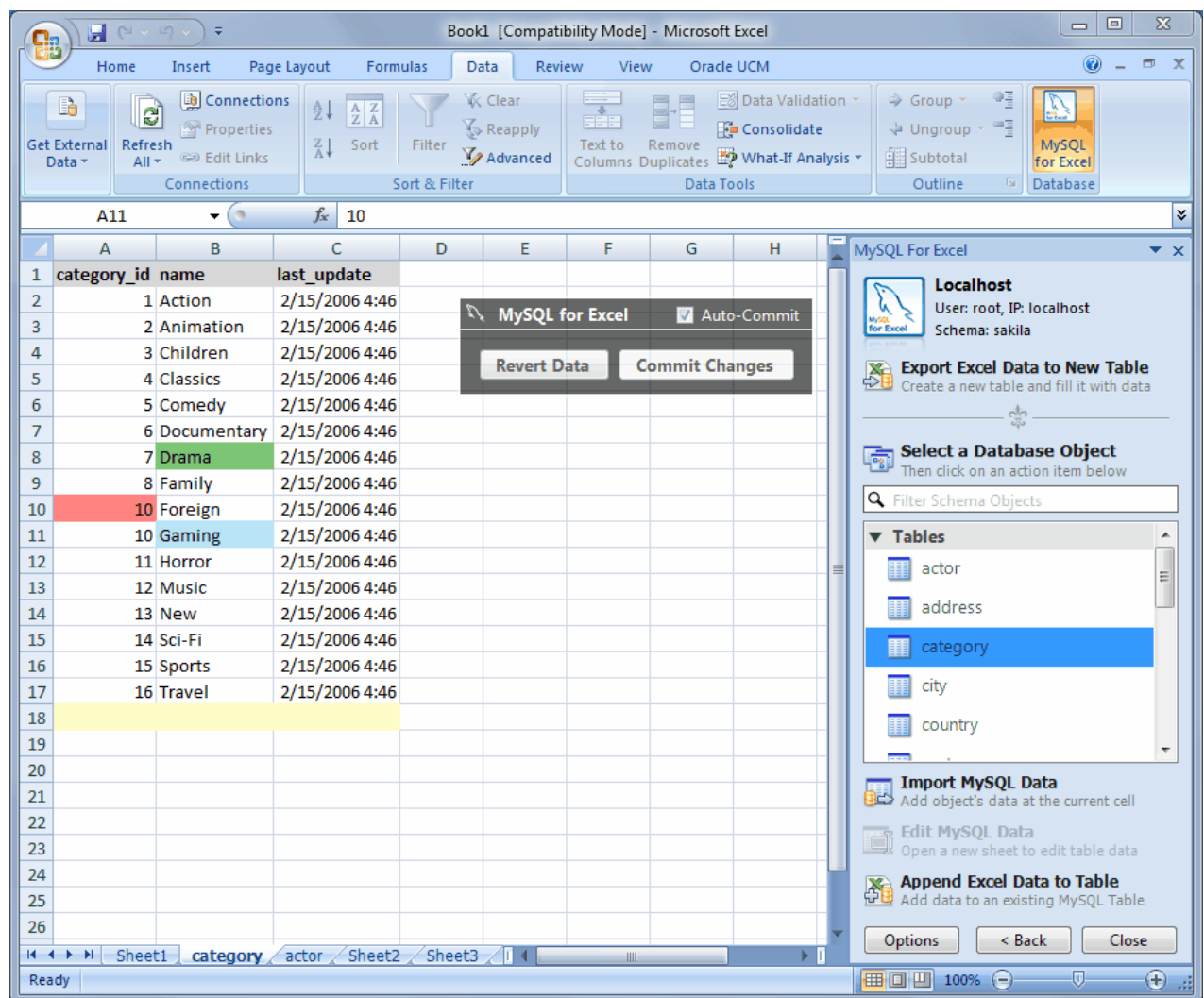
MySQL for Excel enables you to load and edit MySQL data directly from Microsoft Excel. Changes are immediately committed if the **Auto-Commit** option is enabled, or done manually by pressing **Commit Changes**.

The example below uses the `category` table of the example `sakila` database, but the screen will look the same for any table. Within MySQL for Excel, **Open a MySQL Connection**, click the `sakila` schema, **Next**, select the `category` table, click **Edit MySQL Data**, then choose **Import** to import the data into a new Microsoft Excel worksheet for editing.

Note

For additional information about the importing procedure, see [Chapter 9, Import MySQL Data into Excel](#).

Figure 8.1 Editing table data with MySQL for Excel



The background color represents the status of each cell, and there are four distinct colors that are used while editing table data:

Note

The Green and Blue colors were switched in MySQL for Excel 1.2.0.

Table 8.1 Background cell colors

Color	Description
White	Default color for all cells. This is either the original data, or the data after Refresh from DB is clicked.
Green	Cells that were committed with success.
Blue	Cells that were modified but have not yet been committed.
Red	Cells that generated an error when a commit was attempted. An error dialog is also displayed while the commit is attempted.
Orange	Cells that had a commit attempted, but the commit failed due to detected changes from external sources. For example, a different user made a change to a field after it was imported into Excel. This is a feature of Optimistic Updates .
Yellow	Cells that accept new data. Data entered here is inserted into the MySQL table.

In our example, the green "Drama" field was changed and then committed first, then the blue "Gaming" field was changed but not committed, and then **Auto-Commit** was enabled before changing the "9" to a "10" in column 10, which generated an error because this commit would have added a duplicate value as primary key.

Chapter 9 Import MySQL Data into Excel

Table of Contents

9.1 Choosing Columns To Export	81
9.2 Importing a Table	81
9.3 Import: Advanced Options	82
9.4 Importing a View or Procedure	84
9.5 Adding Summary Fields	85
9.6 Creating PivotTables	87

Data can be imported from MySQL into a Microsoft Excel spreadsheet by using the **Import MySQL Data** option after selecting either a table, view, or procedure to import.

9.1 Choosing Columns To Export

By default, all columns are selected and will be imported. Specific columns may be selected (or unselected) using the standard Microsoft Windows method of either **Control** + **Mouse click** to toggle the selection of individual columns, or **Shift** + **Mouse click** to select a range of columns.

The background color of a column shows the status of each column. The color white means that the column has been selected, and therefore it will be imported. Conversely, a gray background means that the column will not be imported.

Right-clicking anywhere in the preview grid opens a context-menu with either a **Select None** or **Select All** option, depending on the current status.

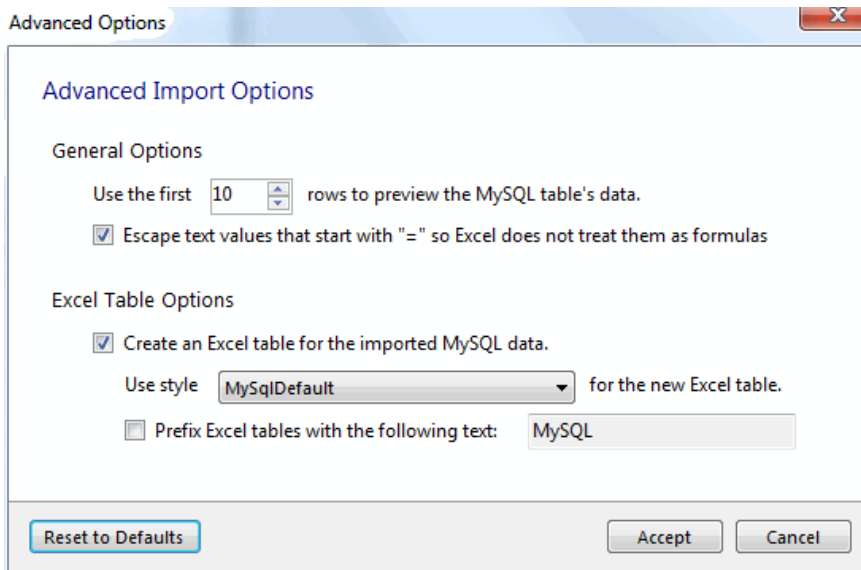
9.2 Importing a Table

The dialog while importing a table includes the following options:

- **[] Include Column Names as Headers:** Enabled by default, this inserts the column names at the top of the Microsoft Excel spreadsheet as a "headers" row.
- **[] Limit to ____ Rows and Start with Row ____:** Disabled by default, this limits the range of imported data. The **Limit to** option defaults to **1**, and defines the number of rows to import. The **Start with Row** option defaults to **1** (the first row), and defines where the import begins. Each option has a maximum value of COUNT(rows) in the table.
- **[] Add Summary Fields:** Disabled by default, this option adds a summary field to each column. For additional information, see [Section 9.5, "Adding Summary Fields"](#).

9.3 Import: Advanced Options

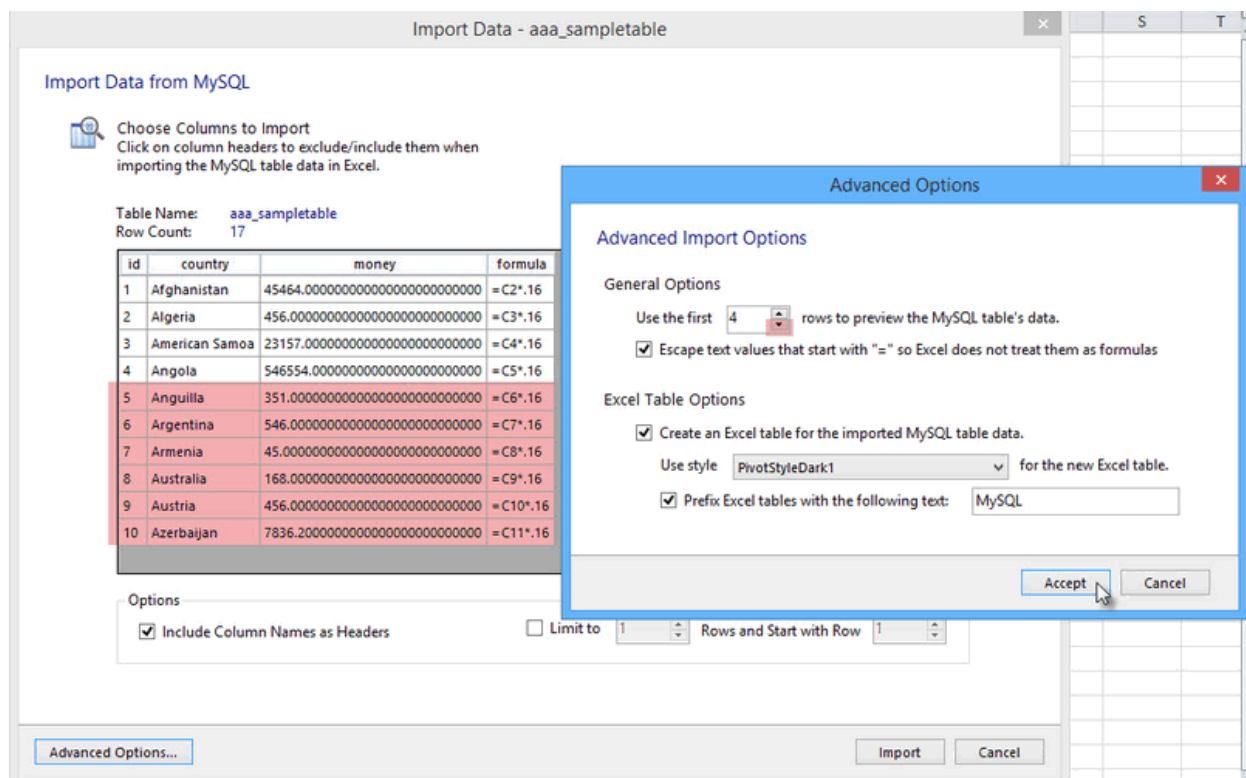
Figure 9.1 Importing table data with MySQL for Excel: Advanced options



General Options:

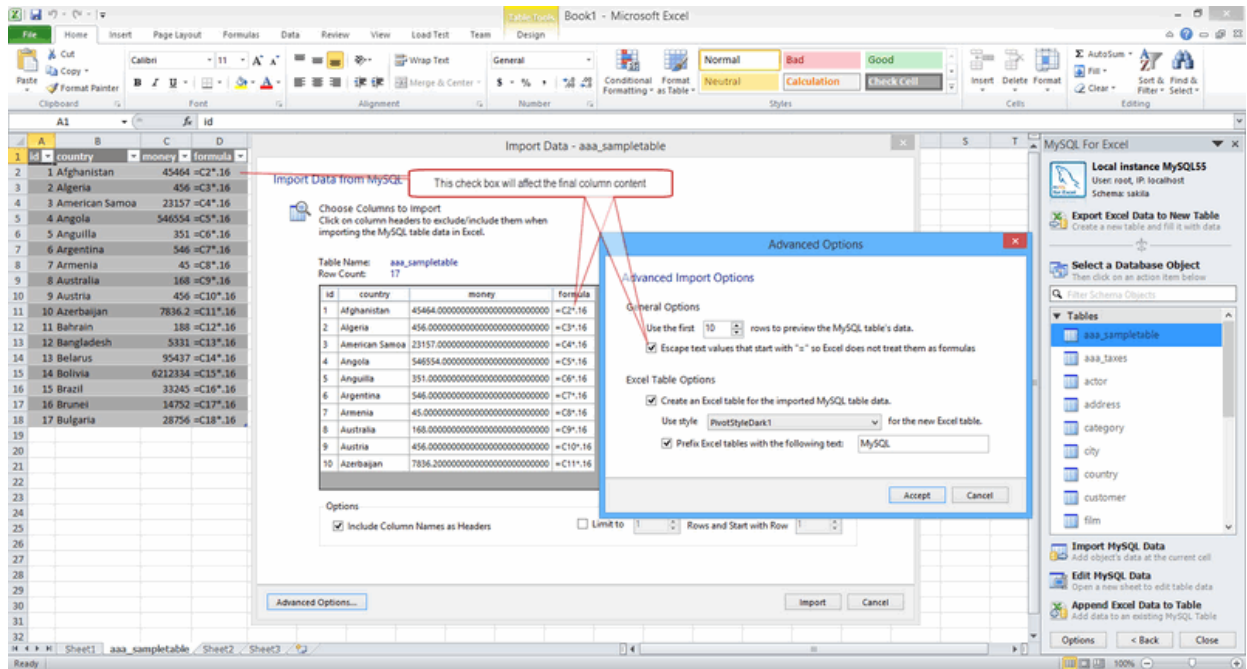
- Use the first [10] rows to preview the MySQL tables data. This affects the preview step in the import process, and defaults to 10.

Figure 9.2 MySQL for Excel: Preview



- ☐ Escape text values that start with "=" so Excel does not treat them as formulas, and is enabled by default. This option may not reflect any differences in the preview because it is only applied after the data is imported into the Excel Worksheet.

Figure 9.3 MySQL for Excel: Escape "=" (formulas)

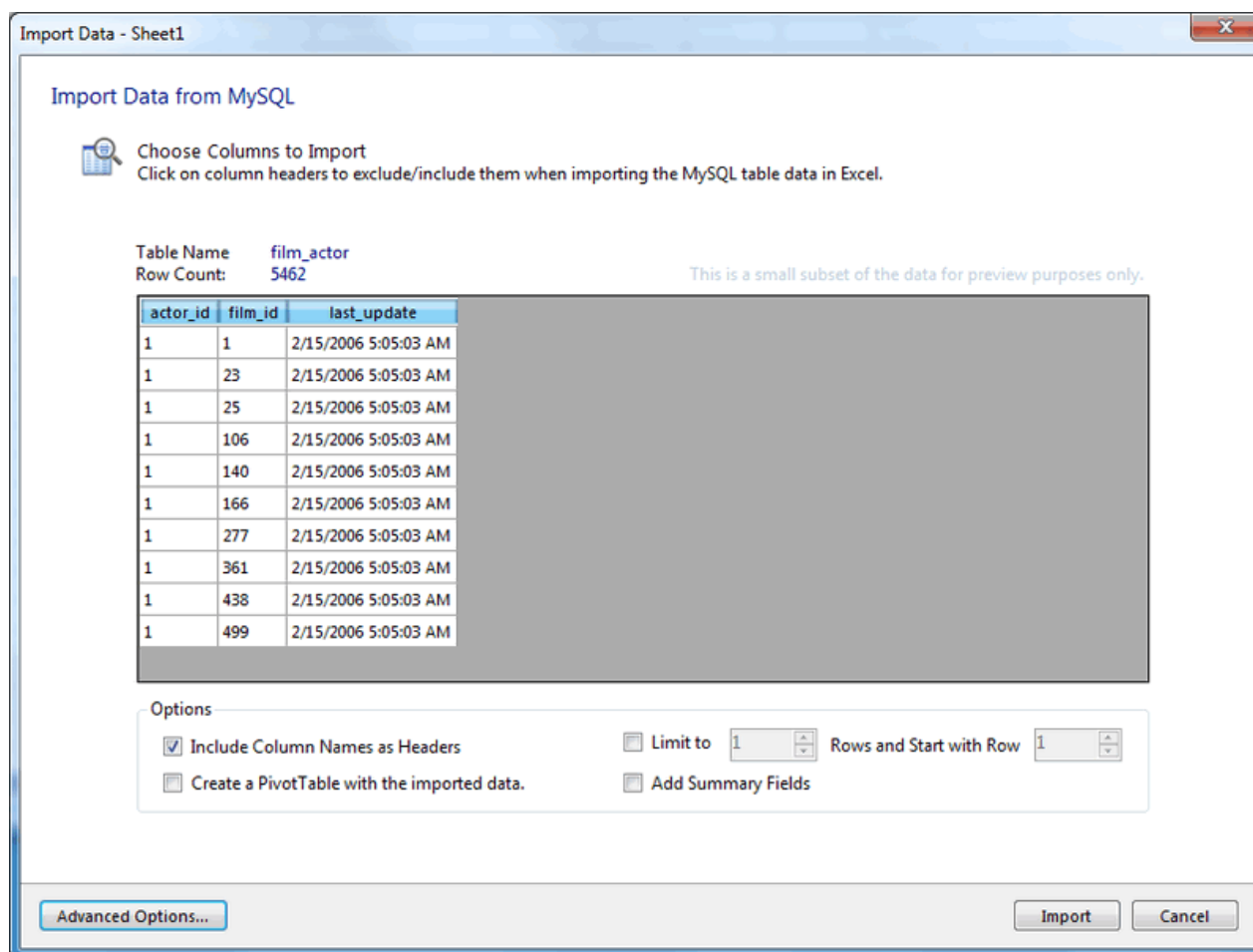


Excel Table Options:

- ☐ Create an Excel table for the imported MySQL table data. Enabled by default.
- Use style for the new Excel table. Defaults to `MySQLDefault`.
- ☐ Prefix Excel tables with the following text: . Disabled by default.

Importing a table displays a dialog similar to the following:

Figure 9.4 Importing table data with MySQL for Excel

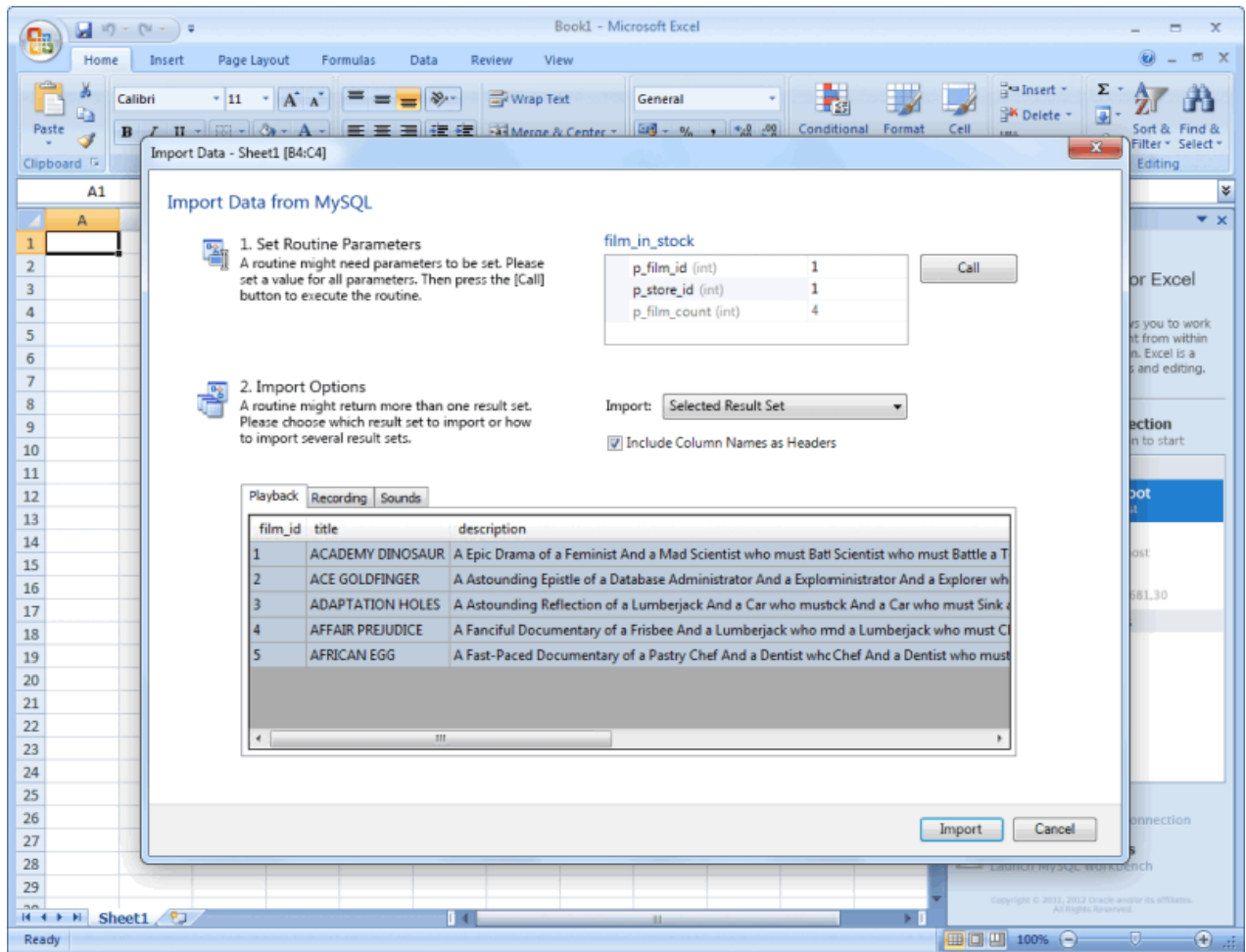


9.4 Importing a View or Procedure

Importing a view or procedure displays a similar dialogue, but with the following options:

- **Include Column Names as Headers:** Enabled by default, this will insert the column names at the top of the Excel spreadsheet as a "headers" row.
- **Import:** Because a procedure might return multiple result sets, the import options include:
 - **Selected Result Set:** Imports the selected tab sheet. This is the default behavior.
 - **All Result Sets - Arranged Horizontally:** Imports all result sets into the Excel Worksheet horizontally, and inserts one empty column between each result set.
 - **All Result Sets - Arranged Vertically:** Imports all result sets into the Excel Worksheet vertically, and inserts one empty row between each result set.

For example, a dialogue like the following is displayed after importing a procedure and pressing the **Call** button to invoke the stored procedure:

Figure 9.5 Importing called stored procedure data with MySQL for Excel

9.5 Adding Summary Fields

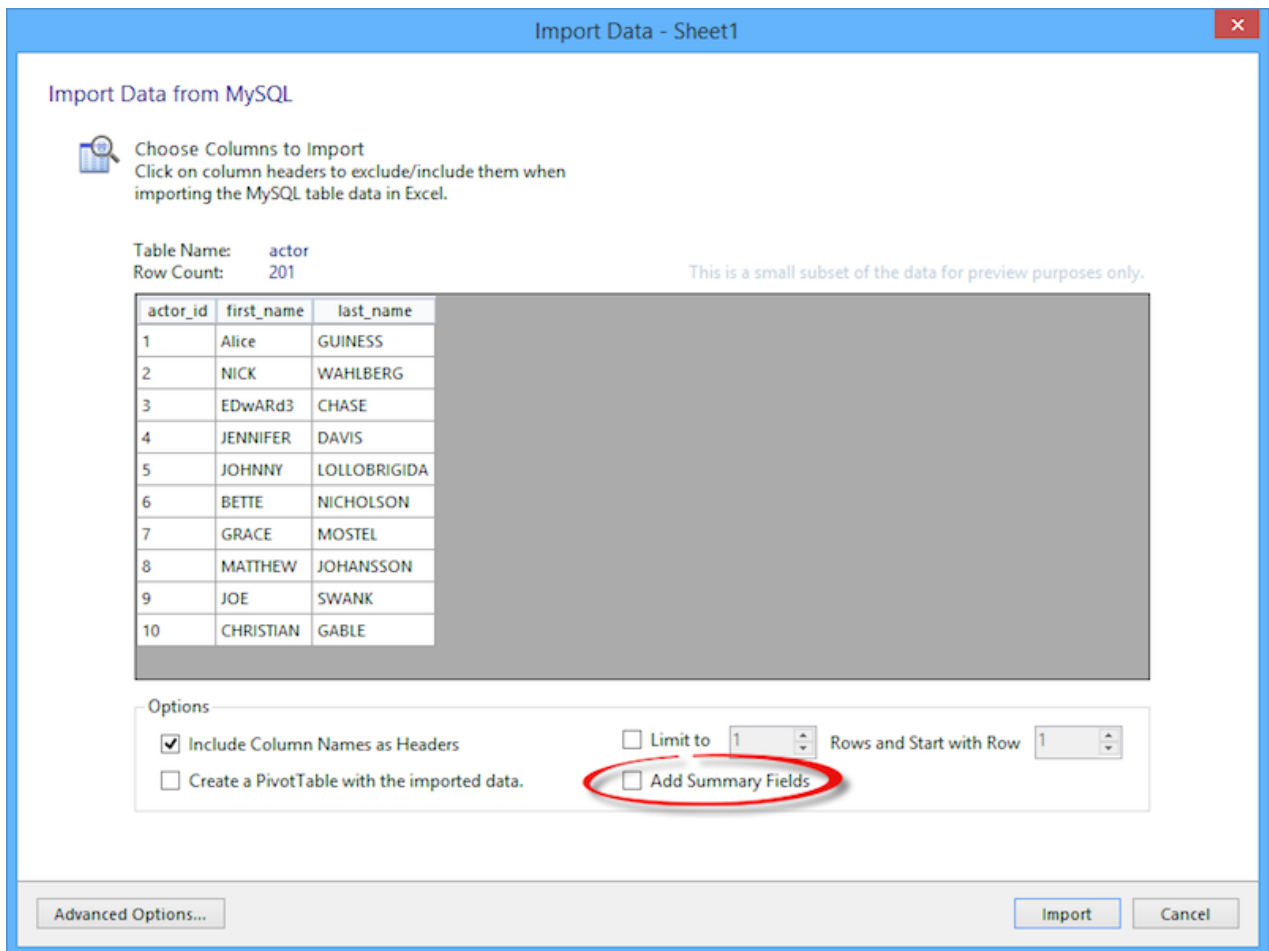
Summary fields are calculated fields, and this option adds summary related functions to each of the imported columns. These fields are added to the last row of the imported table data, and the dropdown of choices includes Average, Sum, Min, and Max.

Note

This feature was added in MySQL for Excel 1.3.0.

The **Add Summary Fields** option (disabled by default) is listed on the import dialog:

Figure 9.6 The 'Add Summary Fields' option



Enabling this option adds a row of summary fields for the appropriate columns in your imported data. Notice the newly created row on the bottom:

Figure 9.7 The new 'Add Summary Fields': the new row

	A	B	C	D
570	4569	999	1	2/15/2006 05:09
571	4570	999	1	2/15/2006 05:09
572	4571	999	2	2/15/2006 05:09
573	4572	999	2	2/15/2006 05:09
574	4573	999	2	2/15/2006 05:09
575	4574	1000	1	2/15/2006 05:09
576	4575	1000	1	2/15/2006 05:09
577	4576	1000	1	2/15/2006 05:09
578	4577	1000	1	2/15/2006 05:09
579	4578	1000	2	2/15/2006 05:09
580	4579	1000	2	2/15/2006 05:09
581	4580	1000	2	2/15/2006 05:09
582	4581	1000	2	2/15/2006 05:09
583	4581			

Select the row to reveal a down arrow, and click it to display a set of summary options:

Figure 9.8 The 'Add Summary Fields' row: choices

4581	4580	1000	2	2/15/2006 05:09
4582	4581	1000	2	2/15/2006 05:09
4583	4581			
4584				
4585				
4586				
4587				
4588				
4589				
4590				

For example, choosing **Sum**:

Figure 9.9 The 'Add Summary Fields' row: sum example

6892
None
Average
Count
Count Numbers
Max
Min
Sum
StdDev
Var
More Functions.

Adjust each summary field accordingly.

9.6 Creating PivotTables

A PivotTable can be created from imported MySQL tables, views, stored procedures, or the entire Excel Data Model.

Note

This feature was added in MySQL for Excel 1.3.0.

An Excel PivotTable report summarizes and provides a visual representation of data in many different ways. It is a native Excel feature, see [PivotTable reports 101](#) for additional information about Excel PivotTables.

Our example covers a simple use case where an empty PivotTable is created from an imported MySQL table. This example uses the "film" table of the "Sakila" database. To create the PivotTable, select the "film" table from the database object's selection panel and then click **Import MySQL Data**. On the **Import Data** dialog, check the **Create a PivotTable** before pressing **OK** to execute the operation.

Figure 9.10 Option: Create a PivotTable with the imported data

Import Data - Sheet1

Import Data from MySQL

Choose Columns to Import
Click on column headers to exclude/include them when importing the MySQL table data in Excel.

Table Name: **film**
Row Count: **1000**

This is a small subset of the data for preview purposes only.

film_id	title	description	release_year	language_id	original_language_id	rental_du
1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a M...	2006	1		6
2	ACE GOLDFINGER	A Astounding Epistle of a Database...	2006	1		3
3	ADAPTATION HOLES	A Astounding Reflection of a Lumb...	2006	1		7
4	AFFAIR PREJUDICE	A Fanciful Documentary of a Frisbe...	2006	1		5
5	AFRICAN EGG	A Fast-Paced Documentary of a Past...	2006	1		6
6	AGENT TRUMAN	A Intrepid Panorama of a Robot An...	2006	1		3
7	AIRPLANE SIERRA	A Touching Saga of a Hunter And a...	2006	1		6
8	AIRPORT POLLOCK	A Epic Tale of a Moose And a Girl w...	2006	1		6
9	ALABAMA DEVIL	A Thoughtful Panorama of a Datab...	2006	1		3
10	ALADDIN CALENDAR	A Action-Packed Tale of a Man And ...	2006	1		6

Options

☒ Include Column Names as Headers

☒ Create a PivotTable with the imported data.

☐ Limit to 1 Rows and Start with Row 1

☐ Add Summary Fields

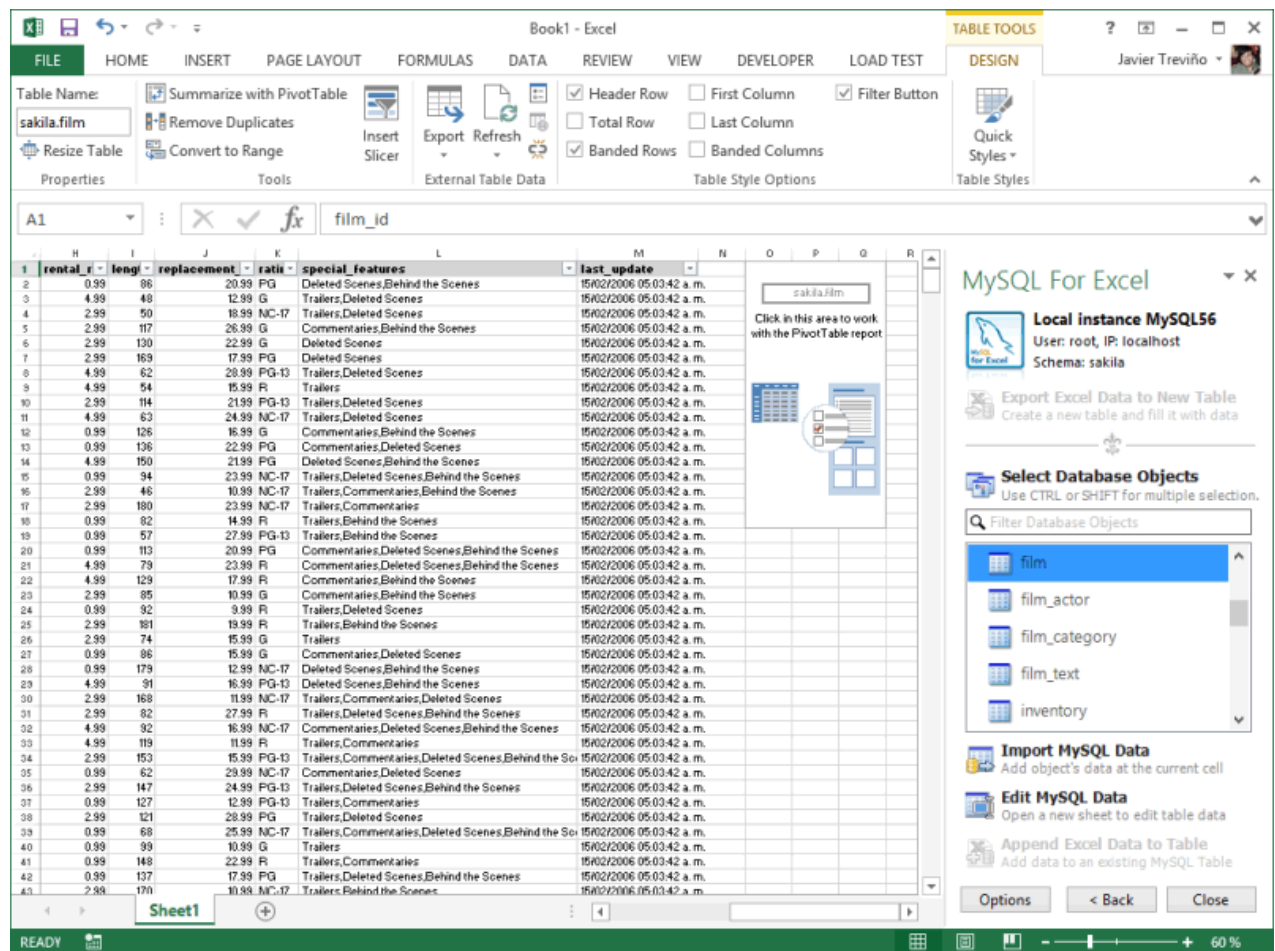
Advanced Options...

Import Cancel

When the **Create a PivotTable with the imported data** option is checked, an empty PivotTable (or a PivotTable placeholder) is inserted just to the right of the imported data. The PivotTable name follows the same naming rules used for Excel tables created from the imported data, but PivotTables can be created with or without enabling the **Create an Excel table for the imported MySQL data** advanced option. That means a PivotTable can be created from an imported Excel range (if the aforementioned advanced option is off), or from an imported Excel table (if the option is on).

Click **Import** to dump the film table's data to an active Excel Spreadsheet, and this also creates a PivotTable for that data.

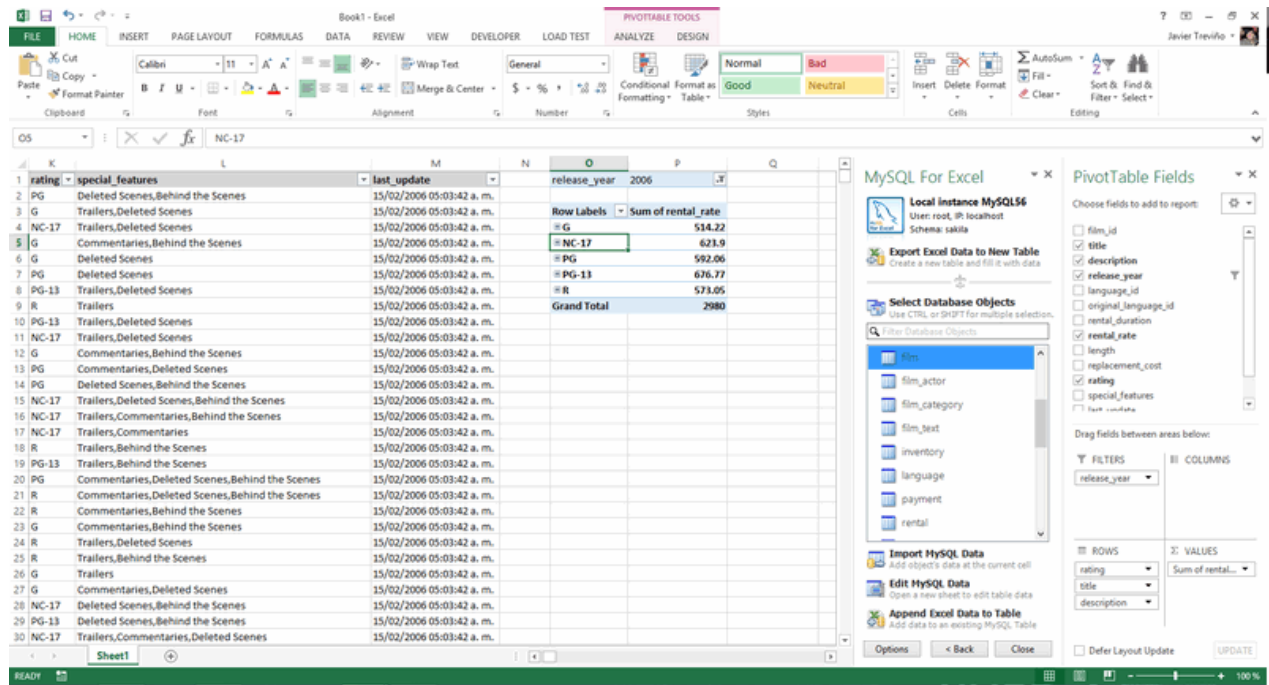
Figure 9.11 PivotTable Example: Empty PivotTable



Clicking the PivotTable opens a PivotTable Fields panel to next to the MySQL for Excel panel, and from here you can select fields you want to summarize in the PivotTable report. Drag and drop fields from the list to any of the FILTERS, COLUMNS, ROWS, or VALUES areas, depending on the visualizations you want in the report. The report is completely dynamic, meaning that you can change the views by moving fields around the areas until you see the visualization you need for your PivotTable report.

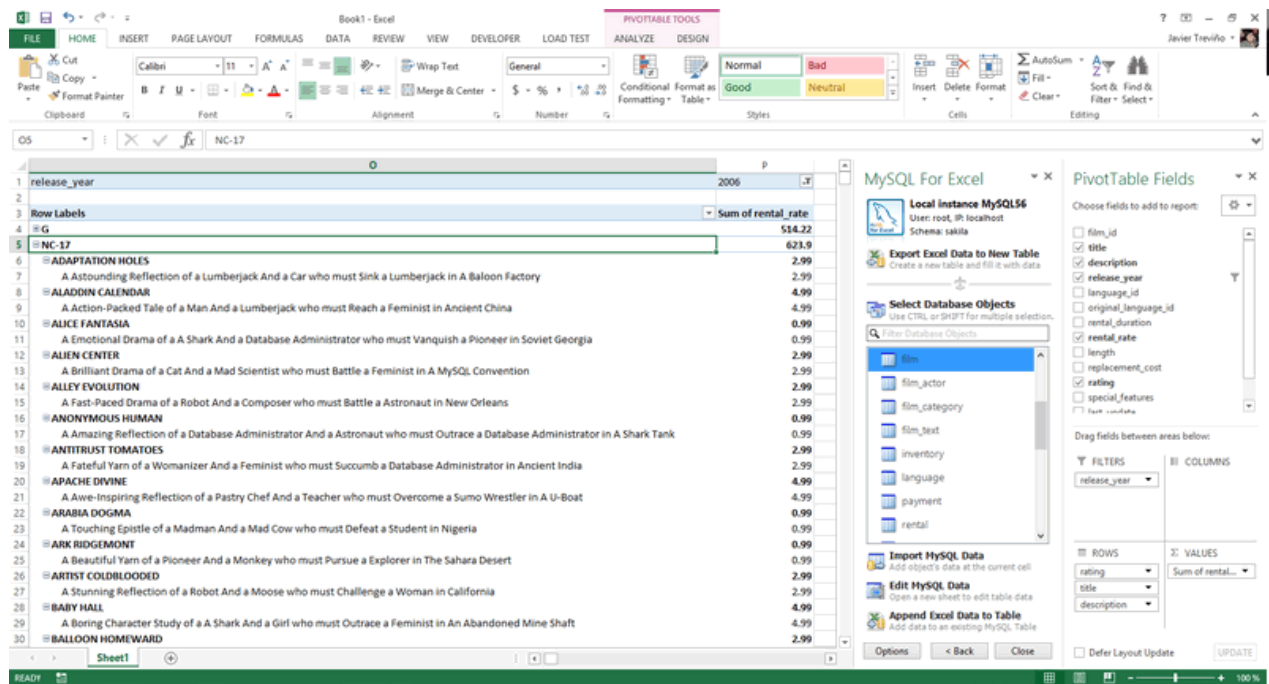
Below is an example PivotTable report using the sakila.film table we imported in our previous example. This report includes a filter by release_year, and it summarizes the rental_rate values while also grouping the data by values in the rating column.

Figure 9.12 PivotTable Example: Film Ratings



Expanding one of the groups reveals its values from the title and description columns.

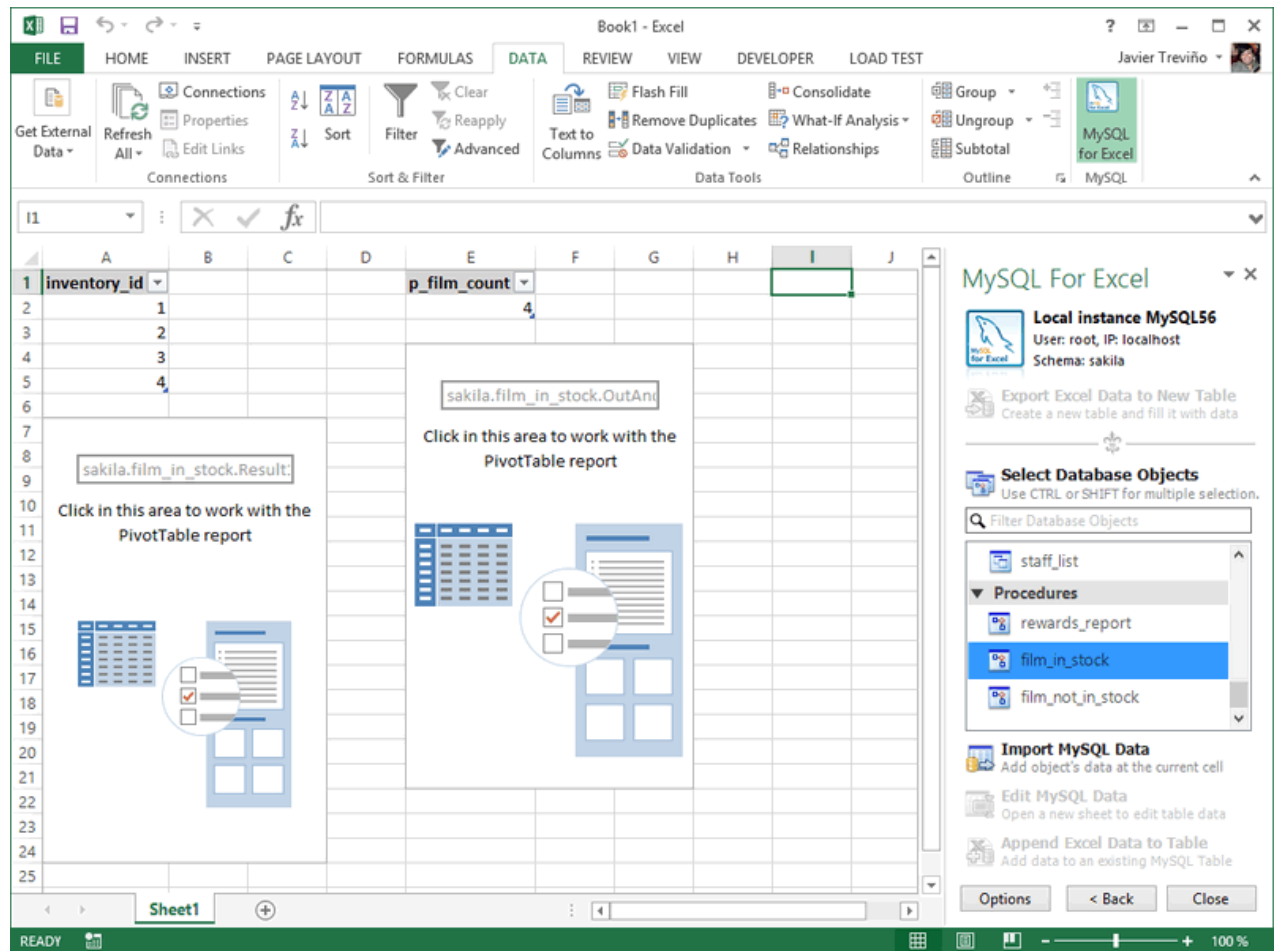
Figure 9.13 PivotTable Example: Expanded Group



We can also do this with data coming from a MySQL view or stored procedures. The only difference is that for stored procedures we can create a PivotTable for each of the imported result sets returned by the procedure's call. Take the following screenshot as an example where we have the film_in_stock stored procedure selected, we configured its input parameter values, and we called the procedure. You can see

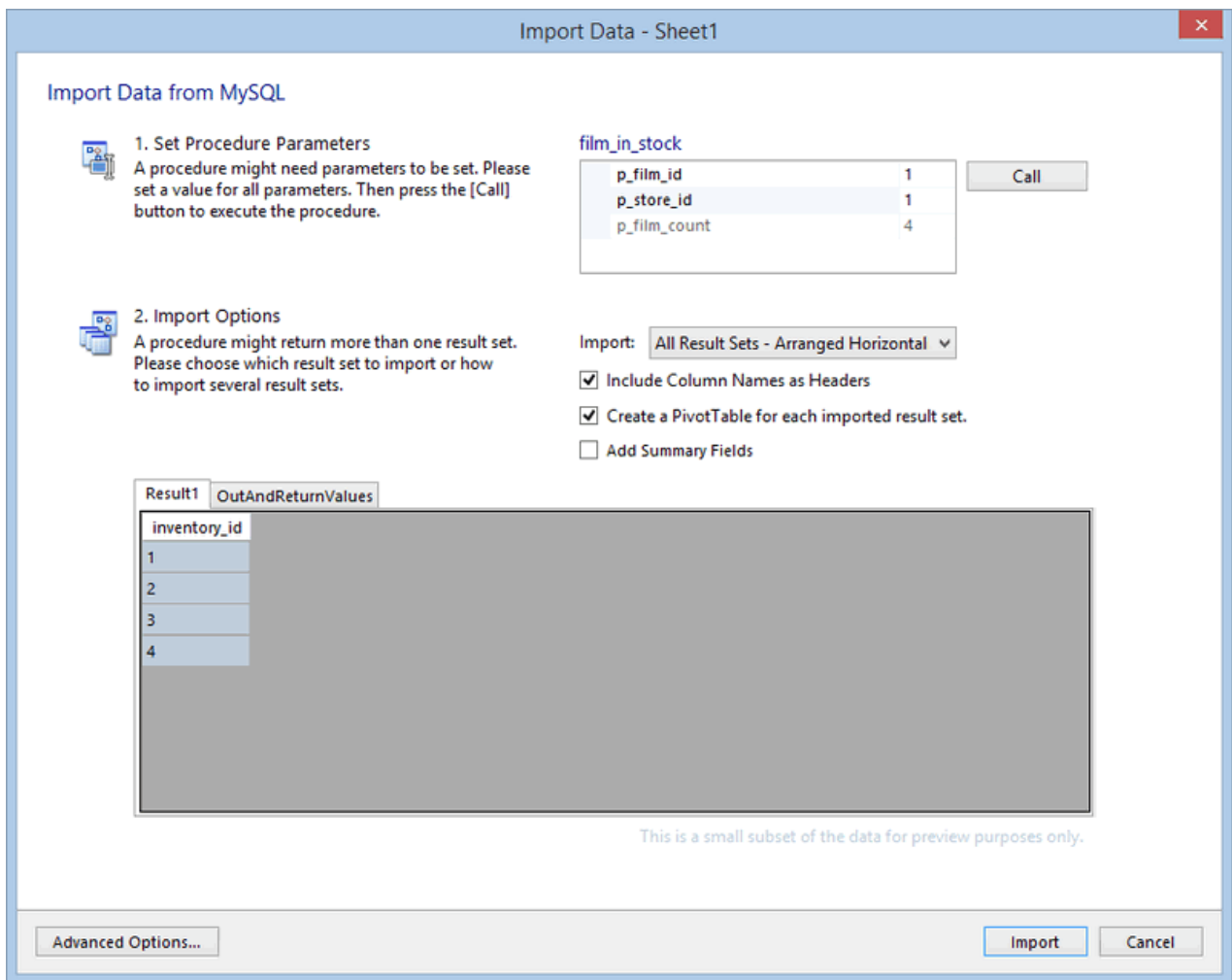
the procedure returned one result set (Result1) and the OutAndReturnValues table (always present if the procedure has output parameters or a return value).

Figure 9.14 PivotTable Example: Stored Procedure



In our example, we selected to import **All Result Sets - Arranged Horizontally**. Because the **Create a PivotTable with the imported data** option was also checked, a PivotTable was created for each returned result set.

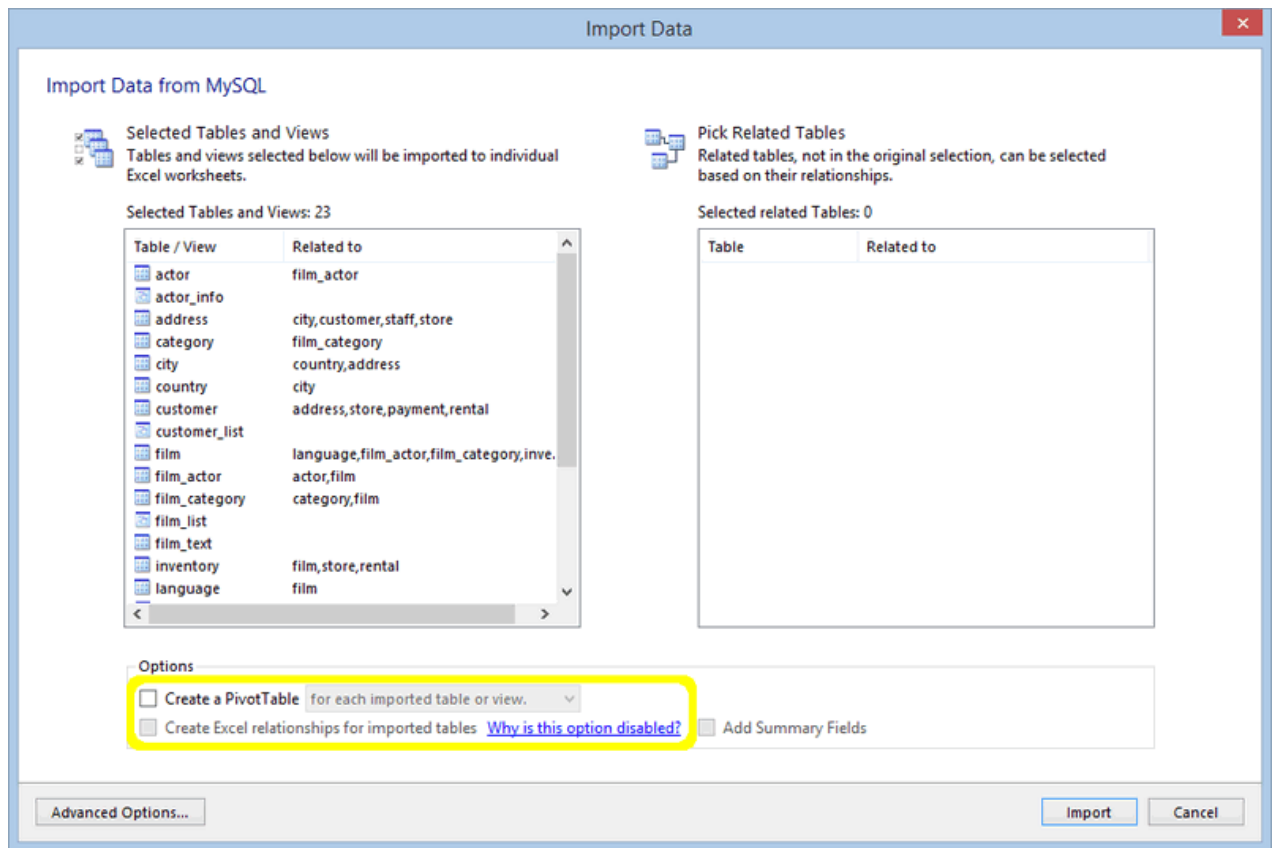
Figure 9.15 PivotTable Example: Arranged Horizontal



An important use case for PivotTables is when we create it for multiple related tables as typically a single table does not contain all of the data needed by a PivotTables report. You can create a single PivotTable tied to the data in the current Excel Data Model that contains fields from several related tables. That way you can use the data in a single report for an entire MySQL schema if needed. However, you can only do this in Excel 2013 (and later) where the Excel Data Model is available.

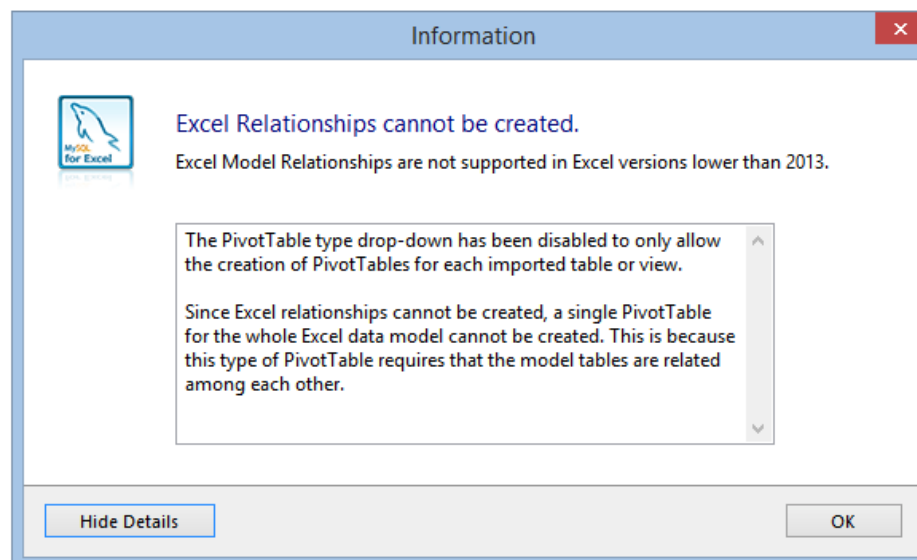
In Excel versions before Excel 2013, only a PivotTable for each imported table or view can be created. This is because a single PivotTable for the entire Excel Data Model requires that the tables are related to each other. If Excel relationships cannot be created, then this type of PivotTable cannot be created. So in these cases, the Import Data dialog looks like the following sample screenshot:

Figure 9.16 Disabled 'Create Excel relationships' option before Excel 2013



Clicking **Why is this option disabled?** displays an information dialog with an explanation of the disabled controls.

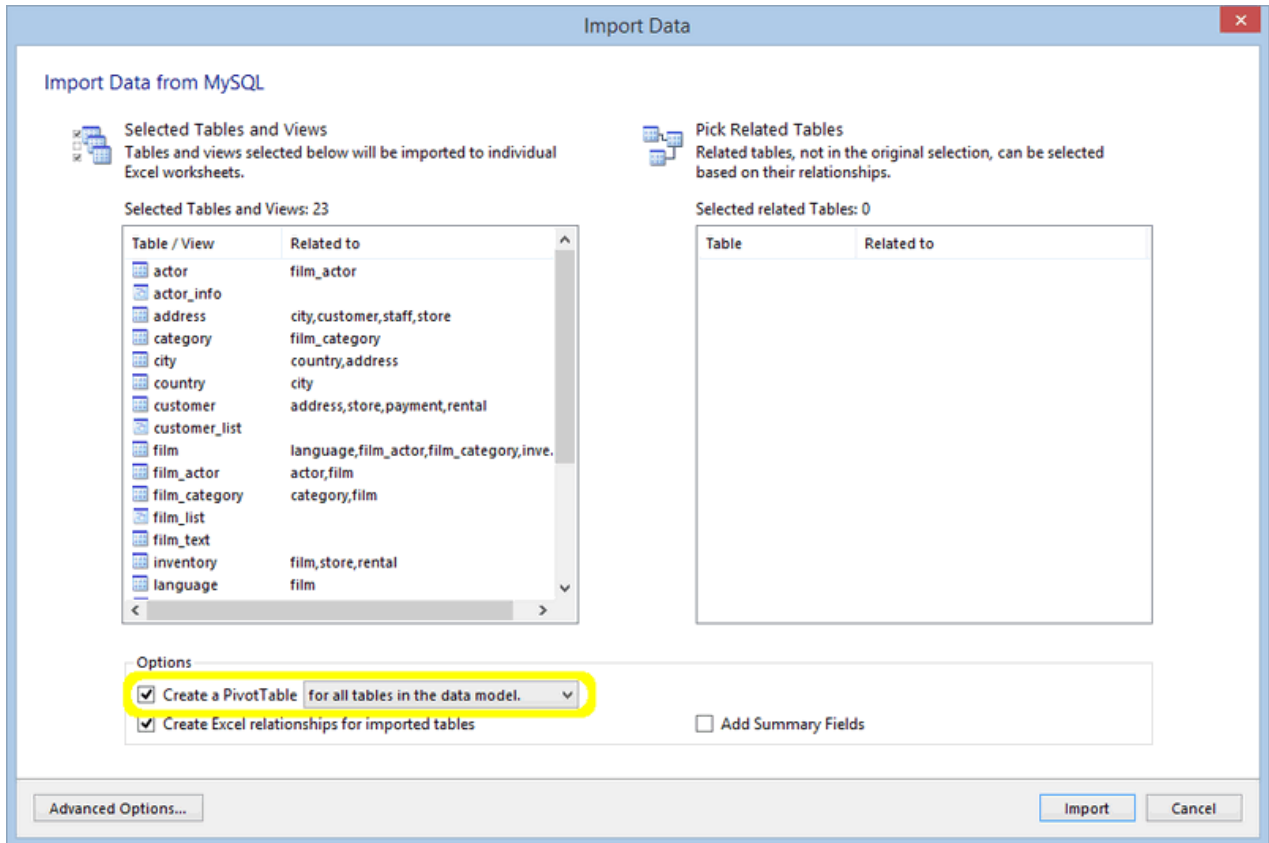
Figure 9.17 Disabled 'Create Excel relationships' option description



Our next example uses all tables in our schema. You can manually choose each table or use **Control + A** in the database objects list to select them all. When clicking **Import Multiple Tables and Views**, the

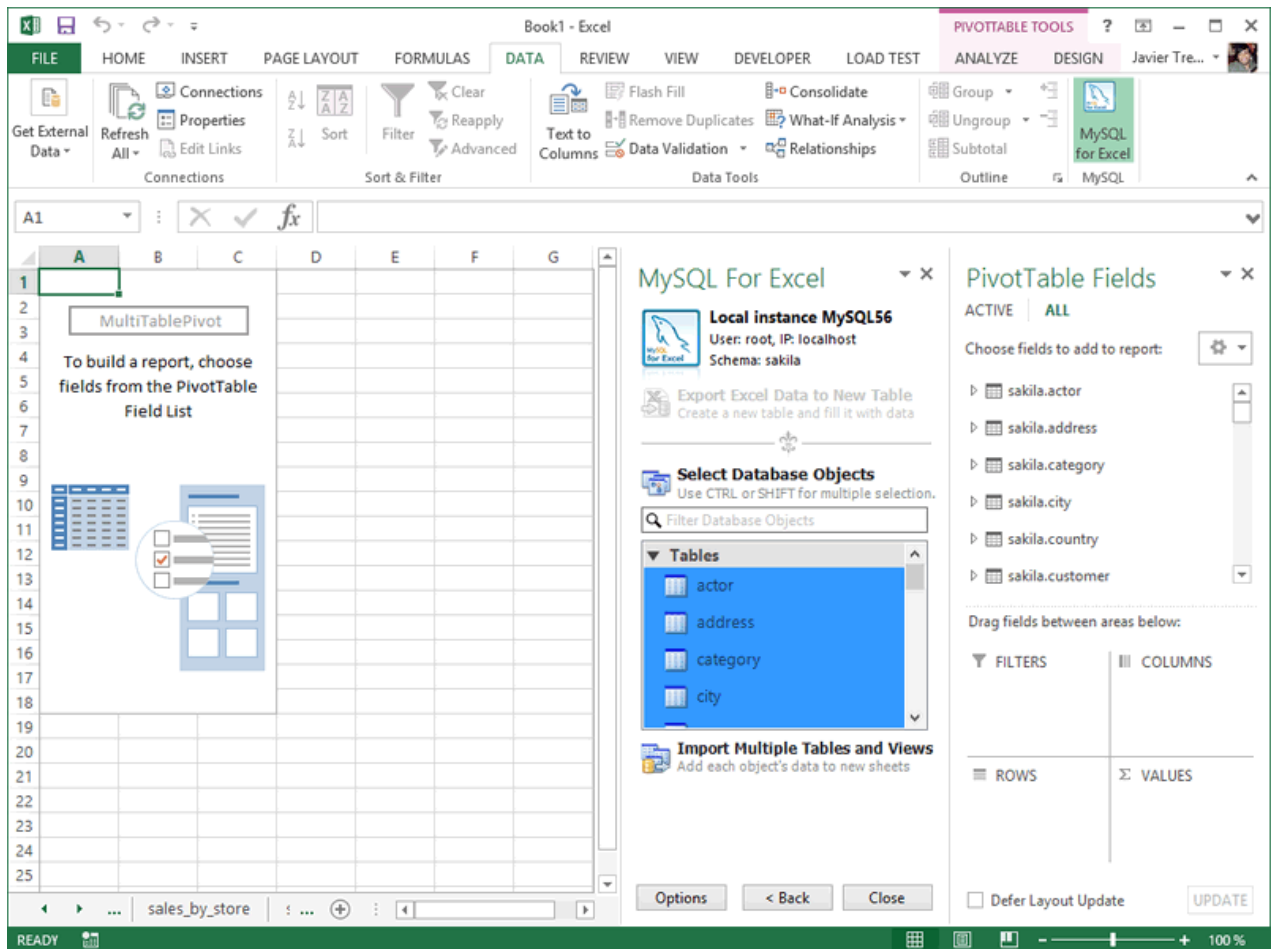
Import Data dialog appears as shown below. We need to check the **Create a PivotTable** option, which by default its drop-down is set to **for all the tables in the data model**. Keep that value.

Figure 9.18 Importing All Tables and Views



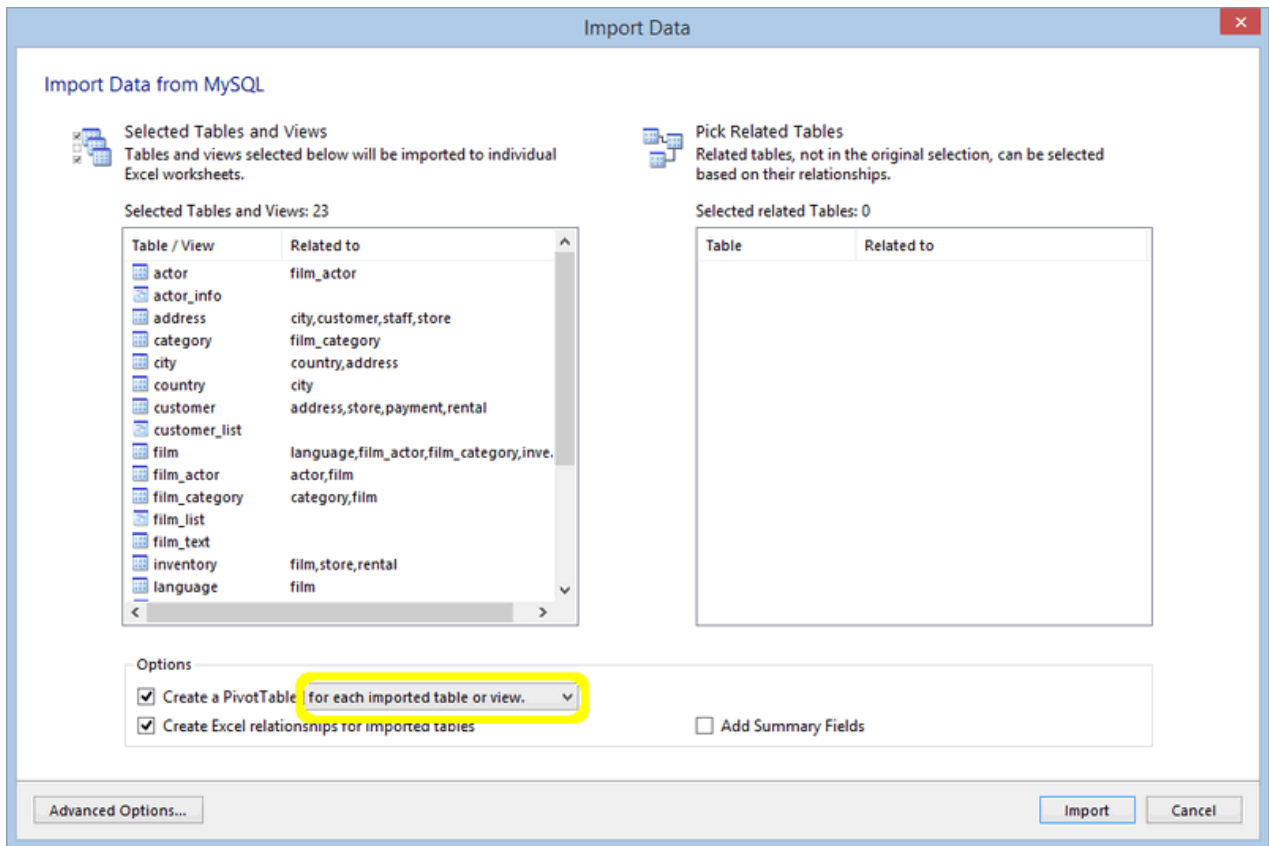
When clicking **Import**, the data in all of the selected tables is imported to Excel, its Data Model and Excel relationships are created, and a new worksheet is created that contains a PivotTable with all of the tables that were imported. This is demonstrated in the screenshot below, and notice that all tables are listed in the **PivotTable Fields** panel.

Figure 9.19 Importing All Tables and Views: Listing



You can also configure the **Create a PivotTable** drop-down list for each imported table or view, which in turn will create a PivotTable for each of the imported tables or views, as opposed to creating a single PivotTable for all of them.

Figure 9.20 Importing Each Imported Table or View



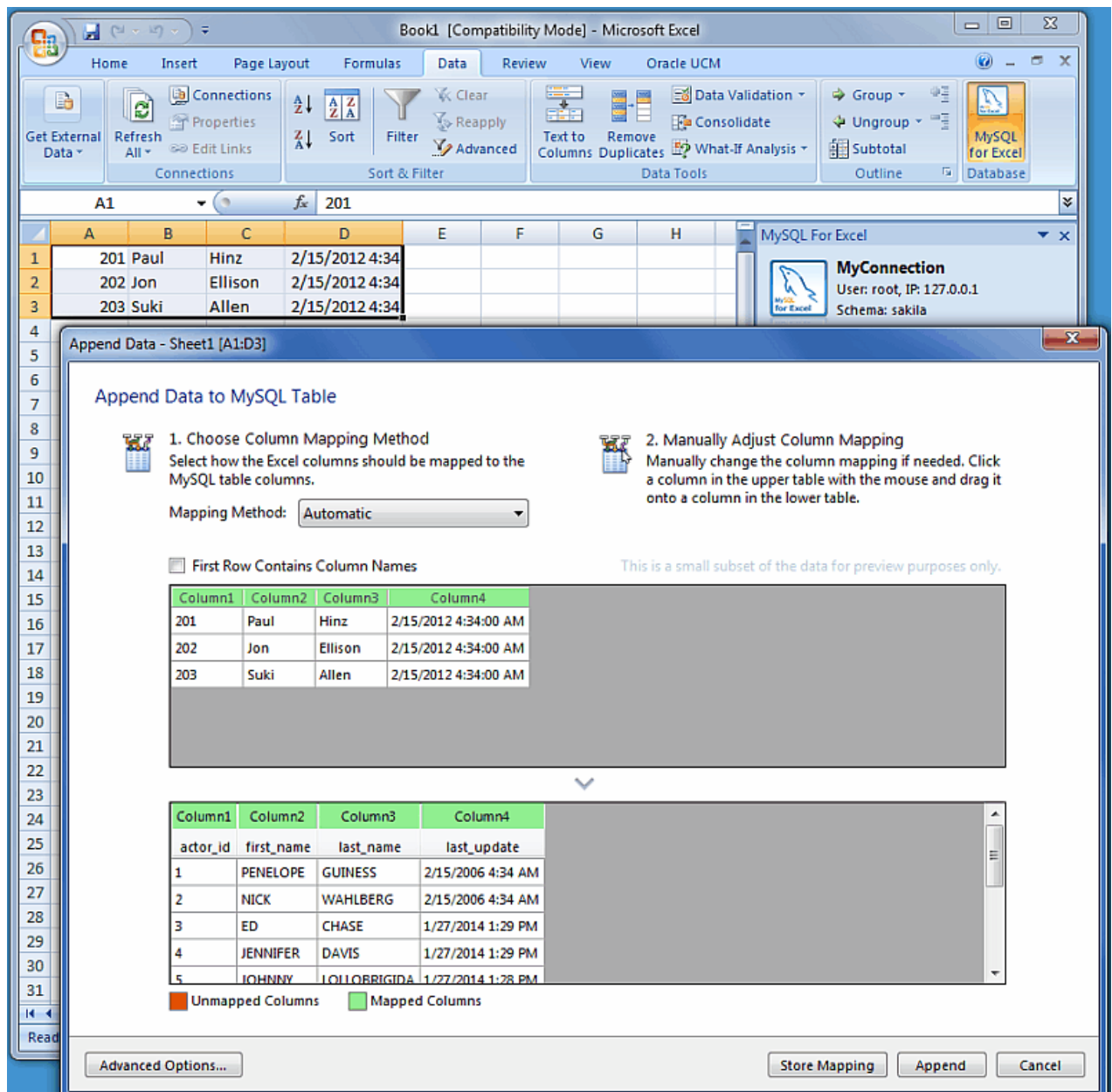
Chapter 10 Append Excel Data into MySQL

Data from a Microsoft Excel spreadsheet can be appended to a MySQL database table by using the **Append Excel MySQL Data to Table** option.

Column Mappings

Mapping the Excel columns to the MySQL columns can be executed automatically (default), manually, or by using a stored mapping routine. An automatic mapping routine is the default, and can be tweaked if every column cannot be matched automatically. The following screenshot shows two columns of Excel data, and the preview dialog after choosing **Append Excel Data to Table**:

Figure 10.1 Appending Excel data to MySQL (Automatic mapping)



General Mapping Information

It is common to tweak the column mappings. A few notes about the manual mapping process:

- Manual mapping is performed by dragging a column from the upper source grid (Excel spreadsheet) and dropping it into the lower target column MySQL table grid. Click anywhere within the column to initiate this dragging routine.
- The color of the header field for each column defines the current mapping status of the column. The colors include:
 - **Green:** A source column is mapped to a target column.
 - **Red:** A target column is not mapped.
 - **Gray:** A source column is not mapped.
- A source column may be mapped to multiple target columns, although this action generates a warning dialog.
- Right-clicking a target column shows a context menu with options to either **Remove Column Mapping** for a single column, or to **Clear All Mappings** for all columns. Dragging a target column outside of the grid removes the mapping.

Mapping Methods

The three mapping methods are described below:

- **Automatic:** The automatic mapping method attempts to match the Excel source column names with the MySQL target table column names. It is then possible to manually tweak the mapping afterwards.

If the automatic process finds zero columns to match, then a simple 1 to 1 matching routine is attempted. Meaning, SourceColumn #1 to TargetColumn #1, SourceColumn #2 to TargetColumn #2, and so on.

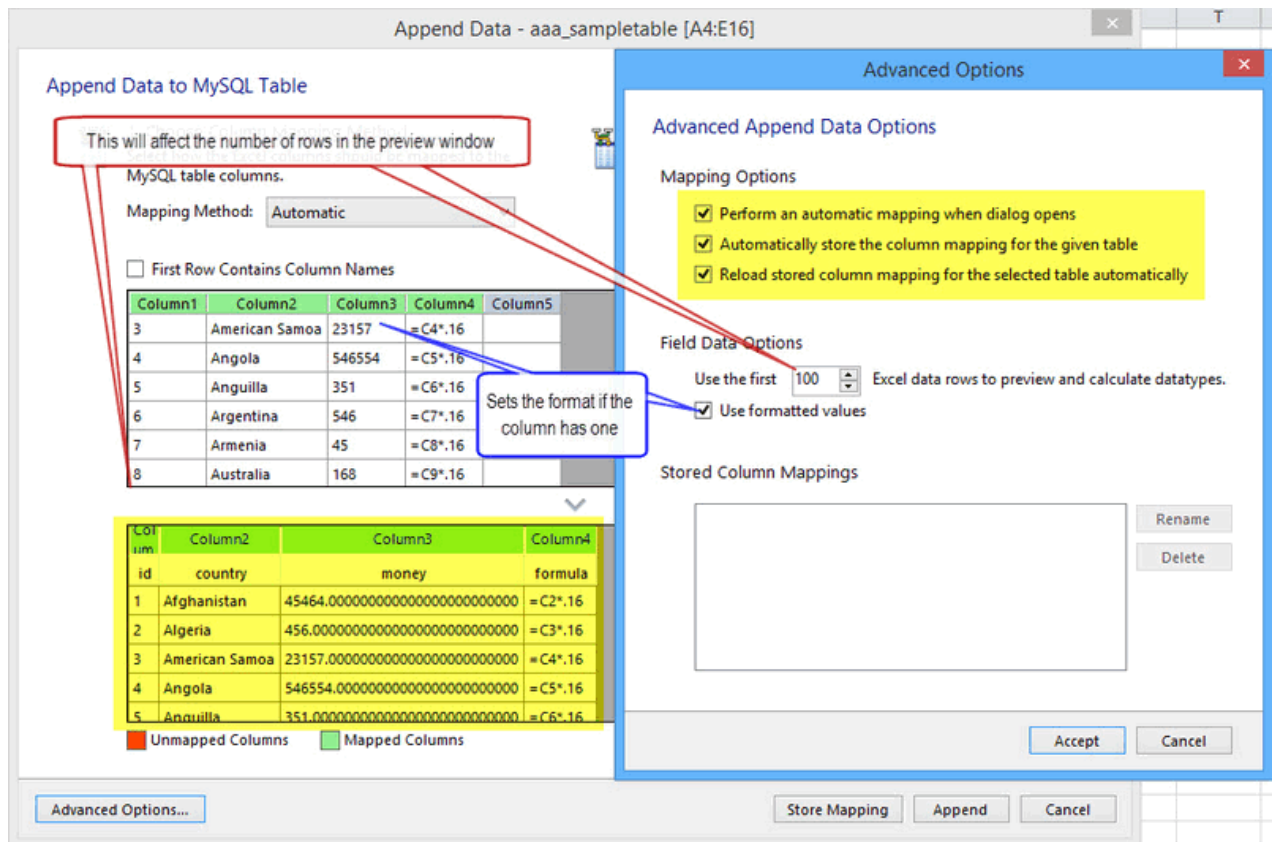
- **Manual:** The source column names are manually dragged (matched) with the target column names. Manual dragging can also be performed after the **Automatic** method is selected.
- **Stored:** Manual mapping styles may be saved using the **Store Mapping** button, which will also prompt for a name and then save it using a "*name* (dbname.tablename)" naming scheme. The saved mapping style will then be available alongside the **Automatic** and **Manual** options.

Stored mappings may be deleted or renamed within the **Advanced Options** dialog.

Append: Advanced Options

There are several advanced options that are configured and stored between sessions for each Excel user. The advanced options are:

Figure 10.2 Appending Excel data to MySQL (Advanced Options)



The advanced **Mapping Options**:

- **Perform an automatic mapping when dialog opens**: Automatically attempt to map the target and source when the **Append Data** dialog is opened. This feature is enabled by default.
- **Automatically store the column mapping for the given table**: Stores each mapping routine after executing the **Append** operation. The mapping routine is saved using the "tablenameMapping (dbname.tablename)" format. This may also be performed manually using the **Store Mapping** button. It is enabled by default, and this feature was added in MySQL for Excel 1.1.0.
- **Reload stored column mapping for the selected table automatically**: If a stored mapping routine exists that matches all column names in the source grid with the target grid, then it is automatically be loaded. This is enabled by default, and this feature was added in MySQL for Excel 1.1.0.

The advanced **Field Data Options**:

- **Use the first 100** (default) Excel data rows to preview and calculate data types. This determines the number of rows that the preview displays, and the values that affect the automatic mapping feature.
- **Use formatted values**: The data from Excel is treated as **Text**, **Double**, or **Date**. This is enabled by default. When disabled, data is never treated as a **Date** type, so for example, this means that a date would be represented as a number.

The advanced **SQL Queries Options**:

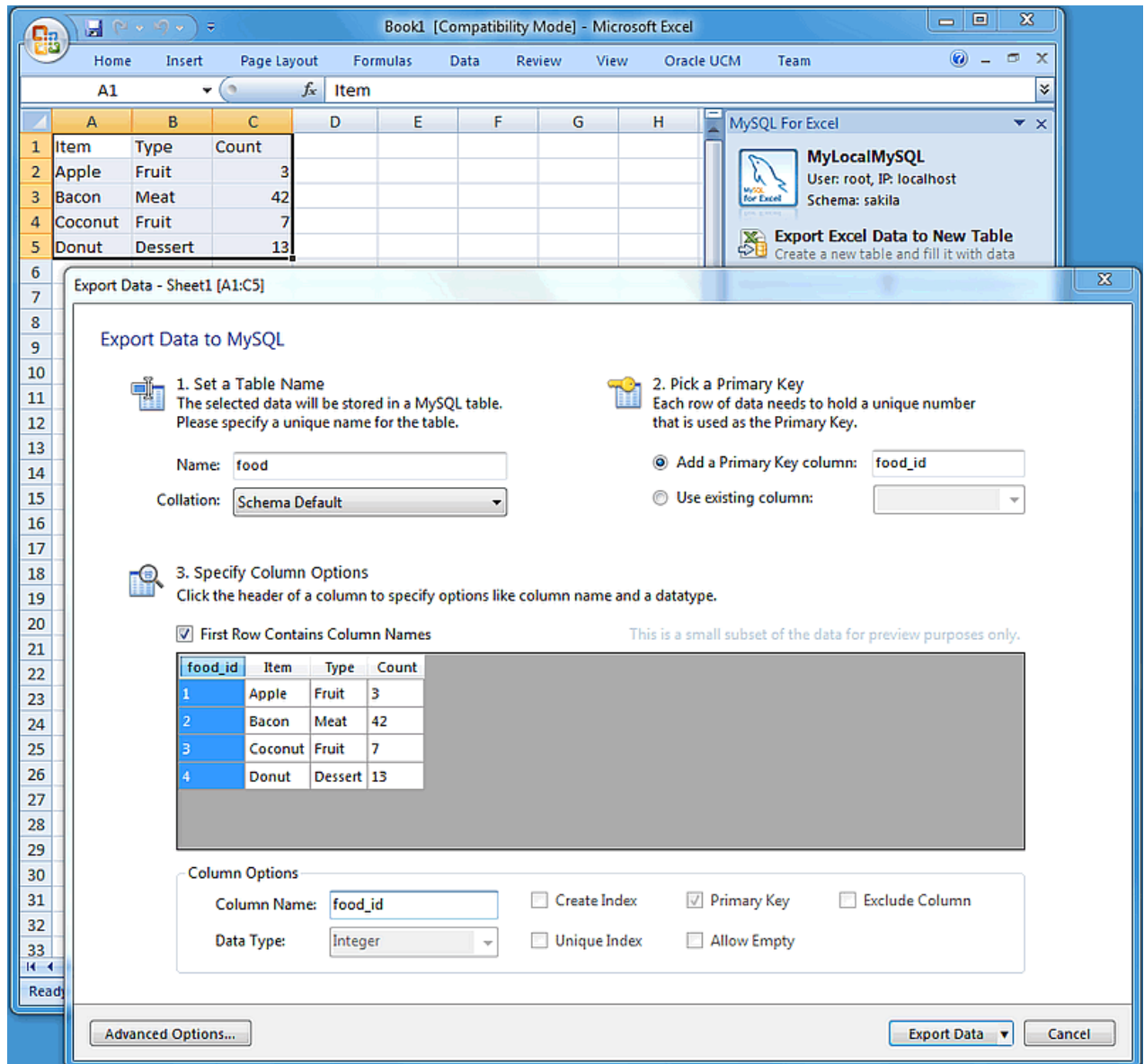
- **Disable table indexes to speed-up rows insertion:** This option is disabled by default, since you must make sure that if unique indexes are present, that the data mapped to that column does not contain duplicate data. This option was added in MySQL for Excel 1.2.1.

The **Stored Column Mappings** is a list of saved column mappings that were saved with the "Automatically store the column mapping for the given table" feature, or manually with the **Store Mapping** option.

Chapter 11 Export Excel Data into MySQL

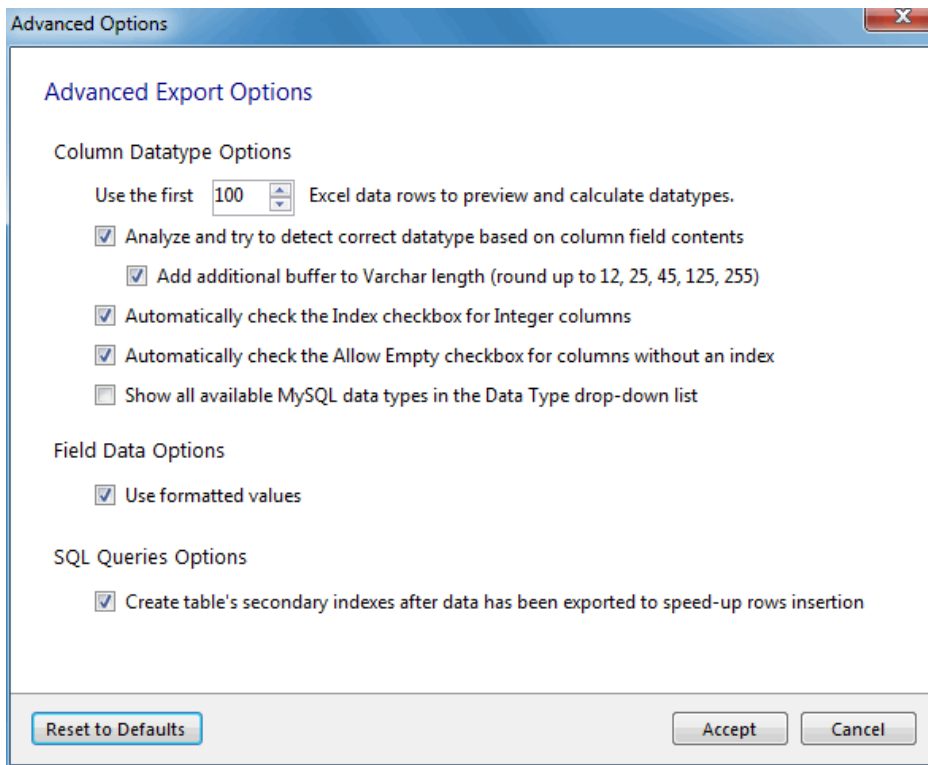
Data from a Microsoft Excel spreadsheet can be exported to a new MySQL database table by using the **Export Excel Data to New Table** option. Exporting data looks like so:

Figure 11.1 Exporting Excel data to MySQL



Advanced Export options

Several advanced options enables you to tweak the exported data. The advanced options dialog looks like so:

Figure 11.2 Exporting Excel data to MySQL (Advanced options)

- **Column Datatype Options:**

- *Use the first 100 (default) Excel data rows to preview and calculate data types:* This determines the number of rows that the preview displays, and the values that affect the automatic mapping feature.
- *Analyze and try to detect correct datatype based on column field contents:* Attempts to analyze the data and determine the data type for the column. The column type is defined as **VARCHAR** if it contains multiple types.
- *Add additional buffer to VARCHAR length (round up to 12, 25, 45, 125, 255):* When the data type is automatically detected and is set to **VARCHAR**, then it calculates the maximum length for all rows within the column, and rounds up the maximum length to one of the defined lengths above.

If disabled, then the **VARCHAR** length is set to the length of the longest entry in the Excel spreadsheet.

- *Automatically check the Index checkbox for Integer columns:* If enabled (default), columns with an Integer data type will have the **Create Index** option enabled by default.
- *Automatically check the Allow Empty checkbox for columns without an index:* If enabled (default), columns without the **Create Index** checkbox checked will automatically enable the **Allow Empty** configuration option.
- *Show all available MySQL data types in the Data Type drop-down list:* By default, only the most commonly used data types are displayed. Enable (disabled by default) this setting to see a list of all MySQL data types.

Note

This option was added in MySQL for Excel 1.3.0

- **Field Data options:**

- *Use formatted values:* When enabled (default), the data from Excel is treated as **Text**, **Double**, or **Date**. When disabled, data is never treated as a **Date** type, so for example this means that a date would be represented as a number.

- **Other options:**

- *Create table's secondary indexes after data has been exported to speed-up rows insertion:* This saves disk I/O for bulk inserts (thousands of rows) since re-indexing will not happen every time a row is inserted, but only once at the end of the data insertion. This option is enabled by default.

- **Note**

- This option was added in MySQL for Excel 1.2.1.

- **Note**

- The following option was **Removed** in MySQL for Excel 1.2.1. Now, the default behavior is to always remove empty columns from the calculations.

Remove columns that contain no data, otherwise flag them as "Excluded": If enabled, columns without data in Excel are removed and not shown in the preview panel. If disabled (default), these columns will exist but have the **Exclude Column** option checked.

Additional Notes

- Entering "0" into a date column.

Entering the value "0" into an Excel date column will convert the value to "12/30/1899" in MySQL. This is because Excel first translates the value to the minimum date in Excel, which is "1/0/1900", because dates are internally stored in Excel as numbers (the days that have passed since "1/0/1900". However, because "1/0/1900" is not a valid date, the committed value to MySQL will change to "12/30/1899" because the .NET MySQL connector automatically converts "1/0/1900" to "12/30/1899", which is the closest valid date.

Chapter 12 MySQL for Excel Frequently Asked Questions

Questions

- [12.1](#): When I'm using Excel to edit data from a live MySQL database, will my changes overwrite changes made by other users? For example, maybe they used MySQL Workbench to edit the same data at the same time.
- [12.2](#): I installed the MySQL for Excel plugin, but can't find it in Microsoft Excel. How do I start it?
- [12.3](#): I click on **Edit Data** and after importing the table data into Excel, I can't sort or move columns around. How can I do that?
- [12.4](#): When editing a MySQL table's data, the Excel worksheet where the data is dumped is protected. How can I unprotect it?
- [12.5](#): The MySQL Workbench connections that use SSH tunneling appear grayed out (disabled) in MySQL for Excel. How can I use a SSH connection?

Questions and Answers

12.1: When I'm using Excel to edit data from a live MySQL database, will my changes overwrite changes made by other users? For example, maybe they used MySQL Workbench to edit the same data at the same time.

The [optimistic updates](#) feature checks for external edits and notifies you of their existence before committing any changes. If an external edit is discovered, you can then choose whether or not to overwrite their changes. This option is enabled by default and can be disabled (to use pessimistic updates). Disabling this option means external changes will always be overwritten. In other words, the choice is yours.

12.2: I installed the MySQL for Excel plugin, but can't find it in Microsoft Excel. How do I start it?

The MySQL for Excel plugin is automatically added to Microsoft Excel's data menu when it is installed. Look for the MySQL for Excel icon, by default it will be listed on the right side of the main menu.

If it's not there, then you might have to reinstall the plugin. But before doing so, first check if it's listed under "Add/Remove Programs" in Microsoft Windows. If not, then it has not been installed. Next, check the Excel Add-Ins list. For Office 2007 this is found by clicking the Office logo in Excel (top left corner), click **Excel Options**, then select **Add-Ins**. Is MySQL for Excel listed as a COM Add-in? If so, then consider filing a bug report (bugs.mysql.com), or attempt to reinstall the plugin.

12.3: I click on Edit Data and after importing the table data into Excel, I can't sort or move columns around. How can I do that?

In order to maintain the mapping of rows and columns in the Excel Worksheet against the rows and columns in the MySQL table, no alteration is permitted on the worksheet (i.e. sorting, deleting rows, deleting columns). If you need to alter the data there you can do that by right-clicking the **Edit Data** window and selecting **Exit Edit Mode**.

12.4: When editing a MySQL table's data, the Excel worksheet where the data is dumped is protected. How can I unprotect it?

The Excel worksheet is protected to not allow alterations to the order of rows and columns. The password used for the protection is a GUID auto-generated at runtime so that the protection is not violated in any way. If you wish to unprotect the worksheet to manipulate your data, you can do that by right-clicking the **Edit Data** window and selecting **Exit Edit Mode**.

12.5: The MySQL Workbench connections that use SSH tunneling appear grayed out (disabled) in MySQL for Excel. How can I use a SSH connection?

This is a known limitation of MySQL for Excel. MySQL for Excel uses MySQL Connector/NET to connect and communicate with MySQL databases. Connector/NET does not have SSH support, so the behavior will change if Connector/NET supports it in the future.